

# HIDDEN MARKOV MODELS FOR ON-LINE SIGNATURE VERIFICATION

**TIAAN WESSELS**



Thesis submitted in partial fulfilment  
of the requirements for the degree of

**Master of Science**

at

**The University of Stellenbosch**

Promotor: Prof C W Omlin

December 2002

# Declaration

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

TIAAN WESSELS

September 2002

# Abstract

The science of signature verification is concerned with identifying individuals by their handwritten signatures. It is assumed that the signature as such is a unique feature amongst individuals and the creation thereof requires a substantial amount of hidden information which makes it difficult for another individual to reproduce the signature. Modern technology has produced devices which are able to capture information about the signing process beyond what is visible to the naked eye. A dynamic signature verification system is concerned with utilizing not only visible, i.e. shape related information but also invisible, hidden dynamical characteristics of signatures. These signature characteristics need to be subjected to analysis and modelling in order to automate use of signatures as an identification metric. We investigate the applicability of hidden Markov models to the problem of modelling signature characteristics and test their ability to distinguish between authentic signatures and forgeries.

# Opsomming

Die wetenskap van handtekeningverifikasie is gemoeid met die identifisering van individue deur gebruik te maak van hulle persoonlike handtekening. Dit berus op die aanname dat 'n handtekening as sulks uniek is tot elke individu en die generering daarvan 'n genoeg mate van verskuilde inligting bevat om die duplisering daarvan moeilik te maak vir 'n ander individu. Moderne tegnologie het toestelle tevoorskyn gebring wat die opname van eienskappe van die handtekeningproses buite die bestek van visuele waarneming moontlik maak. Dinamiese handtekeningverifikasie is gemoeid met die gebruik nie alleen van die sigbare manifestering van 'n handtekening nie, maar ook van die verskuilde dinamiese inligting daarvan om dit so-doende 'n lewensvatbare tegniek vir die identifikasie van individue te maak. Hierdie sigbare en onsigbare eienskappe moet aan analise en modellering onderwerp word in die proses van outomatisering van persoonidentifikasie deur handtekeninge. Ons ondersoek die toepasbaarheid van verskuilde Markov-modelle tot die modelleringsprobleem van handtekeningkarakteristieke en toets die vermoë daarvan om te onderskei tussen egte en vervalste handtekeninge.

# Acknowledgements

We wish to thank J.G.A. Dolfing from Phillips for the permission to use their signature database for this research. We are in debt to Prof. Ben Herbst from the Applied Mathematics Department of the University of Stellenbosch for his help with initial equipment and suggestions. We thank the students who participated in our own experiments for their time. Frank Lepied also deserves acknowledgment for his willingness to help in supporting the Wacom tablet under X-windows. This research was partially funded by grants from the National Research Foundation and from the Telkom-Siemens Centre of Excellence in Satellite Communications, Speech and Image Processing for which we are grateful.

*To my family and friends for their support and inspiration*

# Contents

<b>1 Introduction</b>	<b>1</b>
1.1 Biometrics . . . . .	2
1.1.1 Overview . . . . .	2
1.1.2 Fingerprints . . . . .	4
1.1.3 Voice Recognition . . . . .	4
1.1.4 Iris Scanning . . . . .	5
1.1.5 Retinal Scanning . . . . .	5
1.1.6 DNA Prints . . . . .	5
1.1.7 Dental Records . . . . .	6
1.1.8 Hand Geometry . . . . .	6
1.1.9 Face Recognition . . . . .	6
1.2 Motivation for Signature Verification as a Biometric . . . . .	7
1.3 Problem Statement . . . . .	8
1.4 Objectives . . . . .	8
1.5 Methodology . . . . .	9
1.6 Accomplishments . . . . .	9

1.7	Thesis Outline . . . . .	9
<b>2</b>	<b>Literature Survey</b>	<b>10</b>
2.1	Preface . . . . .	10
2.2	The Signature Database . . . . .	12
2.3	Data Acquisition . . . . .	15
2.3.1	Digitizers . . . . .	15
2.3.2	Instrumented Pens . . . . .	16
2.3.3	Cameras . . . . .	17
2.4	Preprocessing . . . . .	17
2.5	Signature Modelling . . . . .	20
2.5.1	Dynamic Time Warping . . . . .	20
2.5.2	Hidden Markov Models . . . . .	25
2.5.3	Filters and Frequency Domain . . . . .	27
2.5.4	Artificial Neural Networks . . . . .	29
2.5.5	Statistical Modelling . . . . .	32
2.5.6	Optimal Feature Selection . . . . .	34
2.6	Conclusion . . . . .	38
<b>3</b>	<b>Hidden Markov Models for Sequence Processing</b>	<b>39</b>
3.1	Introduction . . . . .	39
3.2	Markov chains . . . . .	40
3.3	Hidden Markov Models . . . . .	43



3.3.1	Problem 1: Observation Sequence Likelihood . . . . .	46
3.3.2	Problem 2: Most Probable State Sequence (Viterbi Algorithm) . . . . .	48
3.3.3	Problem 3: Baum-Welch Parameter Re-estimation . . . . .	51
3.4	Viterbi Training Procedure . . . . .	53
3.5	State Transition Configurations . . . . .	56
3.6	Duration Modelling . . . . .	57
3.7	Other Possible Extensions . . . . .	59
3.8	Summary . . . . .	59
<b>4</b>	<b>A Signature Verification System</b>	<b>60</b>
4.1	Introduction . . . . .	60
4.2	Signature Acquisition . . . . .	61
4.3	Signature Preprocessing . . . . .	63
4.3.1	Rotational Invariance . . . . .	63
4.3.2	Translation Invariance . . . . .	66
4.3.3	Scaling Invariance . . . . .	67
4.3.4	Acquisition Device Invariance . . . . .	67
4.4	Arc-length Parameterization . . . . .	68
4.5	Hidden Markov Models . . . . .	71
4.5.1	General HMM Initialization Scheme . . . . .	73
4.5.2	Reestimation Training . . . . .	78
4.5.3	Single-variate Signal Memory . . . . .	79
4.5.4	Multi-variate Signal Memory . . . . .	88

4.5.5	Randomly Initialized Model . . . . .	90
4.5.6	Statistical Segment Description Models . . . . .	90
4.5.7	Specialized Segment Description Models . . . . .	93
4.6	Authenticity Decision . . . . .	97
4.6.1	Statistical Distributions . . . . .	98
4.6.2	Signature Likelihood Values . . . . .	99
4.6.3	Threshold Calculation . . . . .	100
4.7	Signature Verification Interface and Database . . . . .	106
4.8	Summary . . . . .	108
<b>5</b>	<b>Performance Evaluation and Comparison</b>	<b>110</b>
5.1	Overview . . . . .	110
5.2	Threshold Decision Functions . . . . .	112
5.3	Likelihood Metrics . . . . .	113
5.4	Stability of Pen-up Information . . . . .	114
5.5	Parametrization . . . . .	115
5.6	Signal Reduction . . . . .	115
5.7	State Duration Restrictions . . . . .	118
5.8	Combination of Discriminative Components . . . . .	118
5.9	Multi-Variate Models . . . . .	124
5.10	Signature Segmentation Models . . . . .	125
5.11	Summary . . . . .	127

<b>6</b>	<b>Conclusions and Directions for Future Research</b>	<b>129</b>
6.1	Conclusion . . . . .	129
6.2	Directions for Future Research . . . . .	131
6.2.1	Weighted Decision Function . . . . .	131
6.2.2	Adaptive Sparse Resampling . . . . .	133
6.2.3	Piecewise Continuous HMM . . . . .	134

# List of Tables

- 5.1 **Performance Results:** This table presents a summary of the performance levels attained during the verification experiments described in this study. . . . . 112
- 5.2 **Performance Results:** This table presents the error levels achieved by the unreduced single-variate combination signal memory experiments described in Section 4.5.3. The experiments were conducted on both time and arc-length parametrized signatures where pen-up sections were left intact and subsequently removed. The EERs are calculated from the entire set of forgeries, i.e. all types. . . . . 113
- 5.3 **Parametrization Comparison:** This table shows the forgery rejection ability for the components under an arc-length and time parametrization. . . . . 116
- 5.4 **Performance Results:** This table presents the results for HMMs trained on data reduced by resampling on sparser intervals. It also shows the effect of narrow and wide parameter settings for the models. . . . . 117
- 5.5 **Performance Results:** This table presents the results for HMMs with state duration restrictions disabled and enabled. . . . . 118
- 5.6 **Component Discriminative Power:** This table presents the discriminative power of the individual components as described in Section 4.5.3 seen from different viewpoints. Percentages were rounded to the nearest percentage point and taken at the point of EER equal to 0.6% for the model deduced from arc-length parametrized data with pen-up sections removed. . . . . 119

5.7 **Component Correlation Matrix:** This table presents the correlation amongst the individual components as far as concurrently rejected forgeries are concerned. The percentages are forgery detection rates for FRR less than 1%. . . . . 121

5.8 **Component Overlap Difference Matrix:** This table presents the difference between the correlation percentage and the percentage detected by the component heading the column. . . . . 122

5.9 **Combined Component Performance Matrix:** This table presents the percentage of forgeries which will be detected by combinations of two components. The percentages are forgery detection rates for the FRR less than 1%. . . . . 123

5.10 **Performance Results:** This table presents the results for combinations of different components with strong discriminative characteristics as taken from table 5.9. . . . . 123

5.11 **Performance Results:** This table presents the error levels achieved by the multi-variate signal memory experiments described in Section 4.5.4. For comparison, the result of the equivalent randomly initialized models are provided as well as described in Section 4.5.5. The experiments were conducted on both time and arc-length parametrized signatures where pen-up sections were left intact and subsequently removed. The EERs are calculated from the entire set of forgeries, i.e. all types. . . . . 126

5.12 **Performance Results:** This table presents the error levels achieved by the statistical segment descriptor experiments described in Section 4.5.6. The experiments were conducted on signatures where pen-up sections were left intact and subsequently removed. The descriptors in this experiment are independent of the parametrization. The EERs are calculated from the entire set of forgeries, i.e. all types. . . . . 127

**5.13 Performance Results:** This table presents the error levels achieved by the specialized segment descriptor experiments described in Section 4.5.7. The experiments were conducted on time parametrized signatures only where pen-up sections were left intact and subsequently removed. The descriptors in this experiment are independent of the parametrization. The EERs are calculated from the entire set of forgeries, i.e. all types. . . . . 128

# List of Figures

2.1	<b>Automatic Signature Verification System Abstraction:</b> Most ASV systems adhere to the abstraction depicted in this figure. . . . .	11
3.1	<b>Weather state transitions:</b> A fully connected (ergodic) model configuration.	41
3.2	<b>Weather Hidden Markov Model</b> A hidden Markov model adds observation distributions to states of the Markov chain. . . . .	45
3.3	<b>Forward Procedure</b> Path likelihoods of the partial observation sequence up to a certain time are calculated. . . . .	47
3.4	<b>State Sequences</b> The number of possible paths increases as a power of the sequence length. . . . .	48
3.5	<b>Possible state sequences</b> Each state may have a most probable path ending in that state. We take the path with the highest likelihood. . . . .	49
3.6	<b>Forward-Backward procedure</b> The forward $\alpha$ and backward $\beta$ variables are used to calculate the likelihood of a state sequence being in a certain state at a certain time. . . . .	52
3.7	<b>Types of hidden Markov models:</b> Different types of hidden Markov models are defined by the state transition configurations. . . . .	57
3.8	<b>Unit Duration Sub-states:</b> Duration modelling can be approximated by decomposing a HMM state into unit- duration sub-states. . . . .	58

4.1	<b>WACOM Graphical Tablet:</b> This device is used to measure various aspects of the signing process. . . . .	62
4.2	<b>Sampled Signature Components:</b> Positional, pressure and tilt information are reported by the tablet. . . . .	62
4.3	<b>Pen Tilt Rotation:</b> The reported pen tilt is not invariant to baseline rotation. . . . .	66
4.4	<b>Arc-length Parameterization:</b> A signature sampled on equal time and equal arc-length. . . . .	71
4.5	<b>Alignments Under Different Parametrizations:</b> The figure illustrates how an arc-length parameterization tends to fan out the data at regions of high density in the time parameterization of signals. . . . .	72
4.6	<b>Dynamic Time Warping:</b> DTW performs a non-linear alignment between two sequences. The figure shows two raw sequences (top-left) aligned by DTW (top-right) with timing function (bottom-left) and an entire set aligned to reference (bottom-right). . . . .	75
4.7	<b>Signer Consistency:</b> The sampled signature components for an inconsistent signer and a consistent one. . . . .	77
4.8	<b>HMM Sequence Margin:</b> An alignment with the initialized HMM. The vertical control is shown as the standard deviation corridor round the mean of the state observation densities. . . . .	78
4.9	<b>Tangential/Angular/Centripetal Acceleration:</b> The different accelerations involved in a revolving object. . . . .	82
4.10	<b>Derived Feature Signals for Signature of Figure 4.2:</b> (a) Absolute Velocity, (b) Path Tangent, (c) Absolute Acceleration, (d) Tangential Acceleration, (e) Centripetal Acceleration, (f) Tilt Azimuth, (g) Pressure Derivative, (h) Line Thickness. . . . .	85
4.11	<b>Separate and Unified Alignment:</b> This figure illustrates the variations in timing information between different components in a single signature. . . . .	86



4.12 **Pen-up Information:** The figure shows the relative unstable pen-up sections of a signature. . . . . 87

4.13 **A Two-dimensional HMM modelling  $(x, y)$  signals:** The HMM positions observation density functions along the signature trajectory widening and narrowing them as needed. . . . . 89

4.14 **Segmentation of a signature according to different strategies:** A signature is segmented according to inversions in the  $y$  velocity and when the absolute velocity drops below a percentage of the average velocity. . . . . 92

4.15 **Visuals Descriptors:** This figure illustrates five of the descriptors listed in the specialized segment description model for capturing visual information. . . . . 95

4.16 **Segment Descriptors Alignment:** An alignment of some of the specialized segment descriptors. . . . . 96

4.17 **Normal Distribution  $n(\mu; \sigma^2)$ :** The normal distribution occurs frequently in many natural phenomena. This is the observation made by the Central Limit Theorem. . . . . 99

4.18 **Optimal Threshold:** The log-likelihood at which the authentic and forgery distributions intersect is a threshold which minimizes the joint FAR/FRR. . . . . 101

4.19 **Equal-Error Rate:** A trade-off exists between the FRR and FAR. The point of intersection of the curves are called the equal error-rate and often used to report the performance of an ASV algorithm. . . . . 101

4.20 **Standardized Log-Likelihood Histograms:** The plot shows histograms for the log-likelihoods for authentic signatures according to one of the implemented HMM configurations. . . . . 102

4.21 **Non-Central F(11,45,5) Superimposed on Log-Likelihood Histogram:** Maximum Likelihood Estimation is used to fit density functions to data. . . . . 104

4.22 **Discrete Cumulative Distribution Function:** The inverse distribution function is used to map surface area to a threshold. . . . . 105

4.23	<b>Rejection Count Histogram:</b> The plot shows the histograms for the number of individual components rejecting authentic and forged signatures. . . . .	106
4.24	<b>Signature Verification Graphical User Interface:</b> This application allows the user to enter a training signature set and verify a signature for authenticity.	107
6.1	<b>Sigmoidal Function:</b> The plot shows how the steepness of the sigmoidal is regulated by the value of $\lambda$ . . . . .	132
6.2	<b>Path Tangent Reduction:</b> These plots show the path tangent signal deterioration under reduction. . . . .	133

# Chapter 1

## Introduction

Human ill-intent forces society to frequently call for verification of the claimed identity of a person. Throughout history, various personal identification schemes have been devised in an attempt to curb the problem of one person masquerading as another, most often for illegitimate purposes. Historical records show how early inhabitants of the Nile valley routinely employed primitive biometric verification in a number of everyday business situations. Individuals were formally identified by unique physiological parameters such as scars, measured physical criteria or a combination of features such as complexion, eye colour, height and so on. These primitive biometrics were used in agricultural transactions and legal proceedings. Passwords have been and still remain a popular method of proving one's legitimacy to gain access to a certain area whether a middle-age castle or a modern-day computer network. They suffer though from inherent shortcomings in that they may be forgotten or entrusted to another person. Identity documents such as passports also play an important role to authenticate the claimed identity of a person especially in sensitive areas such as border control. They may be stolen, however, or falsified up to a degree where it is almost impossible to distinguish a forgery from the authentic document.

Modern biometrics has been used by forensic experts for more than a century now, to link criminals to crime scenes. Biometrics is, as the term suggests, the science concerned with the measurement of biological characteristics. Many biological features are unique to individuals and can therefore be used to identify or authenticate the claimed identity of an individual,

often with a great degree of confidence. The electronic revolution has found its way also into biometrics, seeking to capture the knowledge of the forensic expert in automated systems so they can be deployed on a large scale in commercial environments. Various different biometrics have been isolated and studied. This document is concerned with handwritten signatures as a biometric. The signature of a person is stored physically in some abstract form in the brain of the signer, and is revealed for measurement by muscular control of a pen. Not only is the appearance of a signature unique to each individual, but also the way in which it is created. Thus, accurate measurements of the signing process can serve to uniquely identify a person. In this thesis, we investigate various aspects involved in the development of an automated signature verification system. In order to better understand the role of signature verification, we first take a look at the notion of biometrics.

## 1.1 Biometrics

### 1.1.1 Overview

The isolation and measurement of biological characteristics play an important role in modern day security procedures as the claimed identity of a person can be authenticated by measuring a unique biological feature of that individual and matching it to a known authentic sample. Moreover, identification of a person can be performed by matching these measurements to an entire database of a known population. Biometrics are preferred over more traditional personal identification number (PIN) oriented means of authentication for a number of reasons. Biometrics requires a person to be physically present at the point of verification whereas a PIN can be entrusted to other persons; thus, a positive identification is not beyond doubt. Whereas in the past, biometric authentication has been carried out by human forensic experts, advances in computing technologies in the last two decades have made the automation of the process possible; they allow deployment of biometric authentication systems in commercial environments apart from their more traditional use in the criminal justice system.

In general terms, all biometric identification systems work in the same way. A user must be enrolled into the system by taking measurements of the specific biological characteristics.

These measurements are processed and converted to a digital representation which is then stored in a database together with supplementary information about the individual such as a personal identification number (PIN). Whenever a user needs to be identified, e.g. when entering a sensitive area or conducting a financial transaction, the scan is repeated and the PIN is entered into the system at a verification terminal. This code is then compared to the code in the database by some algorithm to decide on the authenticity of the claimed identity. The advent of the SMARTCARD has made it possible to store the code on a card which is carried by the user and presented whenever personal identification is required. The code generated by the second scan is compared to the encrypted code on the card which obviates the need for a central user database.

The effectiveness of biometric identification schemes are generally judged by three criteria:

1. the false-acceptance rate (FAR) which is the percentage of authentication attempts deemed to be true but are in fact false,
2. the false-rejection rate (FRR) which is the percentage of authentication attempts deemed to be false but are in fact true, and
3. the required processing time of authentication.

Biometric identification schemes have to deal with a trade-off between the FAR and FRR as they often counteract each other; in an attempt to lower a system's FRR, the allowed variance has to be increased which naturally leads to a higher FAR. The optimal performance point in a system is achieved where the FAR and FRRs intersect; this point is referred to as the equal error rate (EER). The aim of biometric authentication schemes is to achieve the lowest possible EER which often entails developing complex algorithms to process and compare the measured biological features. Developing such algorithms is the topic of many research efforts in the field of biometrics.

Many different biological features can be measured to facilitate person identification. Each of these have advantages and limitations and the choice of identifying features depends largely on the context they will be used in. Common considerations when choosing a biometric

scheme for a certain application include (1) the level of reliability needed, (2) the development and deployment costs, (3) the target population demographics, (4) the target operating environment, (5) the speed of operation, and (6) the susceptibility to forgery.

Next, we discuss some common biometrics for identification in use today.

### 1.1.2 Fingerprints

Probably the most common biometric in use today is fingerprint scanning. Low implementation cost and fairly high recognition performance make this an attractive solution for many person identification applications. The scanning of a fingerprint is performed by detecting heat variations on the finger surface; a sensor builds a map of an individual's finger. This map is unique to each individual which makes it suitable to identification. Other approaches are based on optical imaging or measurement of small electrical variations across the finger surface. The optical imaging approach is most susceptible to forgery as it is the easiest to duplicate from an authentic sample.

The large scale practical implementation of a fingerprint based system in the payout of pension funds in South Africa has revealed some problems with fingerprint recognition. Some individuals possess fingerprint patterns which are inherently difficult to verify by currently available algorithms. For individuals depending largely on their hands to perform their work, recognition performance can be impaired by scars. This suggests that fingerprint recognition is better suited for environments where fingers are less prone to damage.

### 1.1.3 Voice Recognition

Voiceprint identification relies on the unique characteristics of the vocal tract of a person which results in a distinct voice character for individuals. Humans can very often recognize a person over the telephone only by hearing the person speak which reinforces this claim. Voiceprinting has become an attractive solution to the endorsement of telephonic banking transactions. The recognition algorithms are challenged though by variance induced in the speaker's voice due to illness, high noise ratios on telephone lines and the acoustics at the

point of recording.

#### **1.1.4 Iris Scanning**

The iris is the colored ring of tissue surrounding the pupil of the eye. It consists of a unique pattern of features such as striations and freckles; they remain unchanged over a lifetime of an individual and are thought to be impossible to forge. This makes iris scanning a highly effective personal identifier. However, it is not well accepted by users due to the sensitive nature of the eye. Iris scanning is at present used mainly to restrict access to high-tech and high risk security environments.

#### **1.1.5 Retinal Scanning**

The retina is the light sensitive layer at the back of the eye which triggers nerve impulses via the optic nerve to the brain. With retinal scanning, the unique patterns on the retina are scanned by a low intensity light source via an optical coupler. It has proven to be quite accurate but does require the user to look into a receptacle and focus on a given point. This is inconvenient if the person wears glasses or has concerns about intimate contact with the reading device. For these reasons, retinal scanning has low user acceptance although the technology itself can work well. In practice, retinal scanning is used marginally compared to iris scanning.

#### **1.1.6 DNA Prints**

DNA (deoxyribonucleic acid) is the hereditary material found in all body cells. It contains subunits called bases which vary significantly across the population and apart from identical twins, the overall pattern of these sequences is unique for each person. A single cell from a biological sample, e.g. blood, saliva, semen or hair, is sufficient for laboratory analysis to extract a DNA print. However, it is unlikely that it will be used in the near future as a commercial identification scheme due to the complexity of extracting the DNA print by current methods. Its use is restricted to forensics to link suspects to biological trace evidence

found at crime scenes. As a biometric, it is so reliable that courts accept it as irrefutable proof of guilt or innocence of suspects.

### 1.1.7 Dental Records

Dental records of a person can sometimes serve as a valuable identification characteristic. Due to their nature, teeth are less subject to decay than other biological features. In cases where fire has destroyed other biological features beyond recognition, the unique arrangement of an individual's teeth can be used as a last resort to identify the unknown person. Bite marks on victims of criminal abuse can also provide a useful clue to the identity of an assailant.

### 1.1.8 Hand Geometry

Hand recognition systems require users to place a hand palm down into a reader. An infrared source within the reader projects an image of the hand as a silhouette; it is captured by a high-resolution digital camera. The reader computes the widths and lengths of fingers and makes up to 90 other measurements from the captured silhouettes. Hand geometry based systems offers a good balance of performance and ease of use. This methodology may be suitable for large user databases or users who may access the system infrequently and may therefore be less disciplined in their approach to the system. Although it is one of the earliest developed biometric systems, it remains a popular identification solution.

### 1.1.9 Face Recognition

Face recognition inspects a digital snapshot of a person's face in an attempt to verify the identity of the person. The position and size of the eyes, nose and mouth and the overall shape of the face contribute to the decision process. The face of the average person undergoes changes over time due to changing hairdos, weight fluctuations, facial hair growth or removal, and glasses. This variability poses a challenge to face recognition systems. Such systems are currently becoming ubiquitous at large public venues such as sport stadiums to assist authorities to detect the presence of assailants in the crowds and as such have raised some



public concern about invasion of privacy.

## 1.2 Motivation for Signature Verification as a Biometric

Unlike the biometrics discussed above which identify an individual by physical attributes, signature verification measures an action of an individual which can be repeated. As [59] states, a signature contains special stroke sequences which are not used in ordinary handwriting. These shapes evolve from routine and training and from the conscious and unconscious influence of the rule to create a unique and individual signature. Signature verification systems rely on the assumption that a person can reproduce his/her signature fairly consistently: it is difficult for a forger to simultaneously duplicate the overall signature appearance, writing speed, force on the pen tip, and the angle with which the pen are held. [69] confirms this by arguing that imitating either overall shape or dynamics of a signature is achievable, but to achieve both is difficult. The imitator is not likely to construct a similar overall shape of a signature without showing his hesitation in the waveform of the writing velocity.

Visual examination of a signature is unreliable for authentication. Untrained human eyes can hardly analyse detailed writing features [68]. The advent of hardware able to measure writing dynamics (see Section 2.3) opened the way for more detailed measurement of the signing process. In addition to the final signature image, several time varying aspects of signatures can be recorded. The analysis of these signals is called dynamic signature verification.<sup>1</sup>

Handwritten signatures have been used for some time to endorse financial transactions even though little or no verification of the signatures is done. This sets it apart from other biometrics as it is a well-accepted method of authentication. It is therefore a particularly attractive solution for making financial transactions more secure; it can more easily be integrated into existing transaction procedures. Although current signature verification systems are currently not as reliable as some other biometrics such as fingerprints and iris scans, even less than perfect authentication performance can reduce the financial losses incurred by credit card companies due to fraud. Development of commercial products targeted at signature verification such as the technologically advanced SMARTPEN [3] underlines the importance of this

---

<sup>1</sup>Static signature verification is concerned only with the analysis of captured signature images.

biometric. This instrumented pen measures accelerations and pen angles during the signing process. The perception is that there is definite commercial value in developing automated signature verification systems.

Stress, illness, and intake of neuromuscular stimulants can influence the signing process. A signature can also evolve and change over the lifetime of an individual. This dynamic nature of a handwritten signature sets it apart from many other biometrics and poses a somewhat different set of challenges to researchers. This brings us to the problem statement addressed by the work presented in this thesis.

### 1.3 Problem Statement

Signature creation is a dynamic time-varying process which can be measured by modern hardware. This results in a number of possible parallel sampled signals each describing some aspect of the signing process. No writer can succeed in exactly duplicating a signature in successive attempts. This leads to variance in the signal profiles of different signature exemplars of a single writer. These variances need to be understood and captured by a mathematical signature model in order not to misinterpret future signing attempts as forgeries. On the other hand, acceptance of variances in authentic signatures must not lead to acceptance of forgeries beyond a required minimum performance level.

### 1.4 Objectives

The primary objective of this thesis is to apply hidden Markov models to the domain of dynamic signature verification with the hope of creating a signature model with similar or better performance than some other modelling methods proposed in the scientific literature. Furthermore, this thesis seeks to identify problems pertaining to various aspects of dynamic signature verification and attempts to provide viable solutions. We will present results of extensive experiments which prove the feasibility of the proposed solution.

## 1.5 Methodology

Hidden Markov models lie at the heart of our signature verification approach. They are a standard method used in automatic speech recognition [71, 23, 36, 48]; however, they can in principle model any non-chaotic time-varying system. They provide us with a great deal of control over various aspects of a model and have the ability to learn from examples. Given the variation in the consistency of different individuals' signatures, model flexibility is imperative for creating an automated signature verification system with the ability to adapt to the signatures of different users given samples of their signatures. We investigate a number of different semantic models in search of a suitable signature model which lends itself to efficient and effective automated signature verification.

## 1.6 Accomplishments

We have developed and documented a functional signature verification system. The algorithms were implemented under the Linux operating system and make use of a Wacom digitizer tablet for capturing signatures. The system was also benchmarked against a signature database used by other researchers. The performance levels attained by the system are satisfactory and prove the feasibility of applying hidden Markov modelling to dynamic signature verification.

## 1.7 Thesis Outline

This thesis is organized as follows. In Chapter 2, we review the progress up to now within the field of dynamic signature verification. We proceed to explain the theory of hidden Markov models in Chapter 3. This prepares the reader for Chapter 4 where the signature verification method used in this work is described. In Chapter 5, we discuss the performance results attained by hidden Markov models with different semantics. We conclude this thesis with a summary of our work and possible directions for future research.

## Chapter 2

# Literature Survey

### 2.1 Preface

Automatic signature verification (ASV) has been a research topic for quite some time. The first active research period appears to date back to the mid-seventies [72, 20, 26, 34, 61, 31] where the majority of the efforts went into developing special hardware to capture the signing process. The aim of this survey is to provide an annotated summary of the different modelling methods published mainly during the period 1989-1999. We hope it will provide the interested reader with insight into the different aspects involved in developing a signature verification system and serve as an overview of the avenues already pursued within the field. Only dynamic signature verification (as opposed to static signature verification) is considered (see Section 2.3 for an explanation). For other surveys see [40, 30]

From a global point of view, the main issues involved in developing an ASV system are (1) the choice of device to acquire signatures (see Section 2.3), (2) the choice of computing hardware to perform the various tasks involved, (3) the algorithms used to achieve the desired effect (see Sections 2.4 and 2.5), (4) the enrollment and maintenance procedure for signatures (see Section 2.2), (5) the configuration of the test database for R&D purposes which apart from the enrolled authentic signatures also contains forgeries (see Section 2.2), (6) the configuration of the database for a production version of the system and (7) the possible need for networking if the system is to be deployed in a distributed scenario.

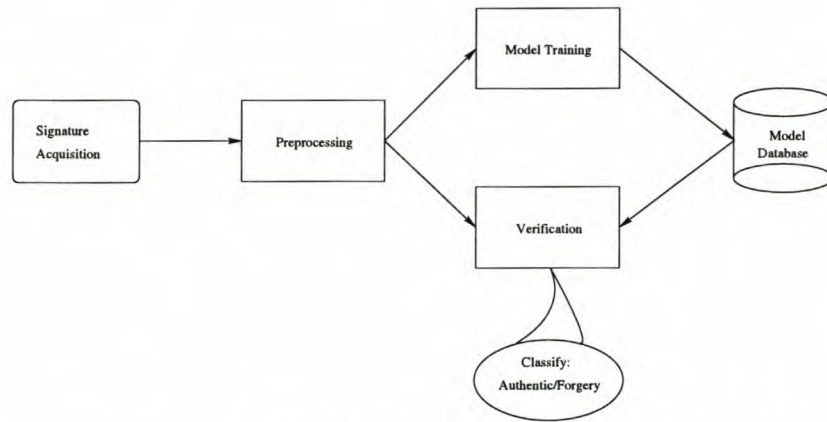


Figure 2.1: **Automatic Signature Verification System Abstraction:** Most ASV systems adhere to the abstraction depicted in this figure.

The functioning of most ASV systems adheres to the abstraction depicted in Figure 2.1. Users enroll in the system by providing a set of their signatures. Signatures are then preprocessed to make them invariant to transformations and to convert them into a format suitable for the modelling process (see Section 2.4). These signatures are then submitted to a modeller which extracts a number of values from this training set which serve as the parameters defining the modelling approach's view of the set. These values are stored in a database along with the necessary details of the user. When the system is presented with a suspect signature claiming to have originated from some user known to the system, the user's model parameters are retrieved from the database and used to decide on the authenticity of the signature according to the semantics of the model. A number of modelling approaches have been applied with varying degrees of success which are covered in Section 2.5.

Performance results reported by the surveyed studies will not be mentioned. Results can be particularly misleading in the field of ASV due to the lack of a standard test bench and the disparate conditions under which the results are produced [51]. As [53] states, differences in the quality and types of forgeries are enough to render any comparison meaningless, as are the differences in the sizes of the training and test subsets, the number of trials permitted and the type of classifier used. Any attempt to compare results of different schemes becomes meaningless unless the results are based on the same data set and the same training/testing partitioning [52]. The credibility of results depends largely on the test database from which the results are deduced. Section 2.2 covers some aspects of importance regarding such a

database as encountered in the literature.

## 2.2 The Signature Database

A benchmark signature database plays a very important part during development of an ASV system. With a well planned database, an algorithm's performance can be gauged and the effect of changes to the algorithm monitored with confidence that one is actually gaining ground. A good database represents a possible real-life deployment scenario as closely as possible. This means that several factors need to be taken into account when creating the database. Various discrepancies between test setups and real world scenarios are revealed in [51] where the author argues that reported results are often an over-optimistic reflection of the performance of the systems were they to be implemented in practice.

Ideally, a system is tested with a large nonhomogeneous population over a long period of time [55]. However, very often databases are reported to have been collected on campuses or in the offices of technical institutions. This contradicts the statistical principle of a population representative sample. Unfortunately, creating such a database is a resource intensive task and to our knowledge there does not exist any database suitable for benchmark purposes which have been donated to the research community.

Factors that need to be taken into account when drawing up a list of users to be enrolled in an experiment are gender, age and dexterity. Various factors regarding the signing environment should also be considered [23, 64]. Users might need to be given time to familiarize themselves with the writing device as it might not have the same feel as an ordinary pen. [19] describes in a fair amount of detail a data collection procedure for creating a signature database. The procedure requires half of the signature set donated by a user to be created in the standing position to ascertain whether there is any significant difference in the two groups of signatures for an individual. For this study, signatures were collected over a four-month period with one or two data-collection sessions per week. Finding volunteers willing to commit to such a lengthy experiment may be difficult. [19] notes that due to the lack of motivation to produce signatures as consistent as possible during an experimental session, signatures might not be of the quality which could be expected in a scenario where the user incurs some penalty

for failing to produce an acceptable signature such as being denied access to a secure area. For this reason, [19] and others offer cash incentives to users to improve the quality of both authentic signatures and forgeries.

Some studies collect all signatures in a single session but it is arguably a more realistic approach to gather the signatures of a user over a longer period of time. This is both to prevent boredom and muscle fatigue *and* to capture natural variations due to physical and psychological changes which is more likely to surface over a longer period of time. In [51], signatures were collected in two sessions at least a week apart. In [53], users provided ten signatures in each of five sessions during one week. In [54], signatures were collected over ten sessions with ten signatures per session. Signatures were collected over a six-month period in [46] which, if time permits, would be handy to determine a more reliable measurement of the true performance of a system in a practical setting. This is because a practical scenario typically requires users to provide a signature set in a single session to minimize inconvenience. Even though there is to our knowledge no study which investigates the variations of signatures over an extended period of time, one can expect a statistically significant change in the signatures of at least a small percentage of users [64, 67]. This means that the actual performance of a system might deteriorate over time because models are built from a set donated in a relatively short time; they do not capturing the variances a user's signature might undergo over time. For this reason then, production quality systems also incorporate adaptive measures for model parameters from authenticated signatures. The assumption being here that users will access a system often enough that authentic signatures will not change so drastically between sessions that they will be rejected. The total number of signatures needed by a model to deduce its parameters, varies among different modelling approaches.

Some studies [18] perform cleanup of the database to rid it from noisy signatures. The criteria used to prune a database include legibility<sup>1</sup>, signing duration within a tolerable distance from the average duration and sabotage such as volunteers signing as Mickey Mouse. The term *goat* in ASV literature refers to a user whose signature has a large negative impact on the overall performance figures of a system [51]. Pruning often seeks to remove such signers. This can result in a false interpretation of performance statistics. Instead, we believe that it is

---

<sup>1</sup>American legislation requires for a signature to be legible

useful when results highlight the number of *goats* as perceived by the particular modelling approach and provide figures with and without the *goats*.

For research purposes, forgeries are very important to measure the performance of a modelling approach. The mere fact that a system accepts authentic signatures is by no means a guaranty that it will reject forgeries. Therefore, the quality of forgeries in a database will to a large extent determine the credibility of results derived from the database. The first kind of forgery often encountered in the literature is the zero-effort forgery also known as a random forgery. This term refers to a signature taken from one enrolled user and presented as the signature of another user. At the very least, a system must be able to reject such 'forgeries' with a great amount of confidence. In [50], the authors state that a system which performs well on random forgeries are likely to perform well on actual forgeries which is a statement open to debate [23]. It does serve a purpose though as [51] points out that credit cards can be stolen while in transit before it is signed by the owner. The forger will in such a case have no idea what the signature looks like. [53] uses only random forgeries for this study which does not attempt to maximize performance but rather serve as a comparison between different modelling approaches. Random forgeries are adequate in such cases as only relative performance is of importance. Some studies [52] show a static image of a signature to forgers and allow them to practice the signature before producing the actual forgery for the database.

Forgeries were collected in [19] by selecting motivated individuals with good manual dexterity and the capability of understanding the basic principals of the system. It was explained that the system inspects dynamic information as well as final appearance. Cash prizes were awarded to the creators of the best forgeries in an attempt to motivate forgers to help create a quality database. The first set of forgeries was created after static images of the signatures to be forged were shown to the forgers. After this, video recordings of the actual signing process were shown to the forgers. They were given three weeks to practice as much as they wanted before submitting the second set of forgeries. As there is no *a priori* knowledge of the forger population expected to attack a dynamic signature verification, this approach seems to be a step in the right direction.

In [55], a modulated sound recording of the signing process was provided to forgers together with the trajectory information of the signature to be forged. Forgers were given time to



practice the signature while listening to the recording after which a set of forgeries is recorded. Again, cash incentives were offered to the creators of the best forgeries.

[23] distinguishes between three types of forgeries (1) home improved, (2) professional and (3) over-the-shoulder forgeries. The home improved variant is created after the forger had only access to a paper copy of the signature. Over-the-shoulder forgeries are, as the name suggests, created after the forger could see the entire signing process of the signature to be forged by standing behind the forger. For the professional forgeries, forensic document examiners provided forgeries based on paper copies of the forged signatures.

From the above, one can understand why it is difficult to compare performance results obtained from different databases. There are simply too many factors which can influence the results; publicly available benchmark databases such as found in the field of speech recognition would be a great advantage to the field of ASV. There is unfortunately some legal aspects involved in releasing the signatures of volunteers for public scrutiny.

## 2.3 Data Acquisition

With static signature verification, only *static* images of signatures are available. This implies that no clue as to the order of signature rendering can be non-trivially deduced from the data. With dynamic signature verification, one or more aspects of the signing process are sampled from a time varying signal. This means that the captured signature can be seen as a time series and well founded modelling techniques can be employed. The acquisition process is very important because the quality of the signals is critical to optimizing the comparison process [40]. Following is a summary of signature acquisition methods reported in the literature.

### 2.3.1 Digitizers

Digitizers (or tablets as they are also known) are at present the most commonly used devices for dynamic signature acquisition [40]. They are often used in Computer Aided Design applications and boast a high spatial resolution for capturing pen movements.

In [23] a Phillips proprietary digitizer called Phillips Advanced Interactive Display (PAID) is used. This device consists of an LCD and orthogonal sensors for pen and finger input sampling. With a sampling rate of 200Hz<sup>2</sup>, the device provides a tuple of x, y, pressure and pen-tilt information with each sample. It should be noted that only the most expensive tablets possess a LCD display on the tablet surface. It is more common for the tablet to use the display of the workstation it is attached to. Other studies employ tablets with differing functionality. In [48, 32], digitizers without the ability to sense pen tilt information are used. There is much variability in the resolution of tablets which may range from 100 to 1000 dpi<sup>3</sup>. Some tablets allow users to use their own pen. As [33] explains, the problem with these are that fingers can protrude into the pressure sensitive area and be registered as part of the signature. Where a special pen has to be used, [33] explains that the pen might not have the same natural feel as 'normal' pens<sup>4</sup> but the quality of the acquired signatures is much better. For a tablet to report pen tilt, a special instrumented pen *has* to be used. In the future, tablets could be used as the man-machine interface for tele-banking systems enabling ASV as the preferred method of transaction authentication [71]. NCR has developed such a signature capturing device for the banking industry [18]. Some tablets have extended functionality normally performed by software such as signal smoothing and compression [51]. The advent of Personal Digital Assistants (PDAs) employing miniature digitizers as the man-machine interface, has spurred renewed interest in the field of handwriting recognition and ASV [54]. To read more about graphical tablets, see [4].

### 2.3.2 Instrumented Pens

A problem with using tablets is their size and cost which seriously hampers their chances of ever finding their way into mainstream ASV applications. Development of specially instrumented pens is an attempt to overcome these problems.

A microprocessor-based interface control card is presented in [49]. A piezoelectric transducer pen is used to convert the signature pressure to an electrical signal before being amplified by a charge amplifier. The output of the charge amplifier is then fed to the interface control card

---

<sup>2</sup>This appears to be a fairly common sampling frequency.

<sup>3</sup>dpi refers to *dots per inch*, the granularity of the pen tip sensing ability on the tablet surface.

<sup>4</sup>This is especially true if the pen is connected to the system by cable as is the case with older digitizers.

to be digitized.

[6] presents a system which employs an instrumented pen with the ability to sense gravitational acceleration. The pen also incorporates a pressure transducer which delivers an electrical signal proportional to the force exerted between the pen and paper. Various problems with accelerometer-based systems and possible solutions are highlighted in [6].

Special pens are less commonly used than tablets for acquisition. The SMARTPEN [3] might change this however. This device has only recently been introduced to the market but is the first serious device dedicated to ASV in a commercial environment. The state of the art technology employs an off-the-shelf ballpoint tip. Sensors producing uncorrelated measurements of forces in three directions exerted on the pen tip is located just behind the tip. It also contains sensors to detect the angles the pen makes with the horizontal plane. On-pen circuitry takes care of data sampling and conditioning. A radio frequency transmitter conveys the signals in secure encrypted form to a base station. This solves the problem of its predecessors which had to be connected to the station by cable. The pen is driven by standard off-the-shelf batteries.

### 2.3.3 Cameras

A fairly new approach to signature acquisition is through the use of a camera [50]. It is argued that cameras are becoming ubiquitous in computing environments and are smaller and easier to handle than digitizers. The tracking of the pen tip during signing is, however, a difficult process and appears to be not as reliable as one would have hoped for. The system sometimes loses track of the pen tip when the signer signs fast. In such a case, the system requires the user to adapt his/her signature to the system. This constraint could result in some difficulty in a commercial environment.

## 2.4 Preprocessing

Preprocessing is an attempt to convert a raw sampled signature to some canonical form by carrying out various operations on the data. [50] assumes that users are consistent in their

style of signing and therefore no normalization is performed: *They write their signatures with a similar slant, in a similar amount of time, with similar dimensions and with similar motion.* Experience has shown that this is an optimistic assumption. A considerable amount of effort was spent on normalization of signature data in several studies (see Section 4.3). This section focuses on some aspects tended to by the various studies examined.

To compensate for differences in the resolution of tablets, [68] linearly normalize x and y-coordinates to reside within a known interval. To compensate for the differences in sampling rates of tablets, [68] uses interpolation to resample signals into a fixed number of points. Various studies [37, 33, 32] report on using cubic smoothing B-splines for interpolation.

Different samples of a writer's signature might be created on differing baselines if the acquisition phase does not restrict the signing action to a uniform orientation. It would be advantageous if such a restriction could be lifted as there is no guaranty that signatures will not deviate from a given baseline for some writers even if provided. [42] disagrees with this by stating that there is no need for rotational normalization if a baseline is provided. Furthermore, one cannot assume that users will sign their signature the same size every time. This imposes the need for an operation which makes signatures scaling invariant. [54] states that rotational differences can serve as a distinguishing feature. Various techniques are employed in the literature to make signatures rotational and scaling invariant which are summarized here.

In [67], a signature is normalized by finding a smallest enclosing circle for it. The center of this circle is selected as a reference point to convert the signature into polar coordinate form, i.e.  $(r_t, \theta_t)$ . Once in this form, the  $r_t$  component is normalized with respect to the radius. The  $\theta_t$  component is normalized by subtracting the value of the previously sampled point i.e.  $\theta_{t-1}$ . Rotational and scaling invariance is thus achieved by creating a sequence  $(\frac{r_{t+1}}{r_t}, \theta_{t+1} - \theta_t)$ .

In [71], rotational invariance is achieved by regarding a signature as a sequence of vectors in the two dimensional Cartesian plane. Each vector is normalized by subtracting the angle the very first vector makes with a principal axis. A potential problem with this approach is the dependence on a single vector for normalization. This might degrade performance if a signature is unstable in the starting sequence of a signature.

[65] maximizes the variance of a signature with respect to the x axis. After initial rotation of the signature, a least squares regression line is fitted to the x data sequence to decide on a further 180° rotation. The net angle is then used to normalize the two pen tilt components sampled from the tablet<sup>5</sup>. This approach does not work well for signatures which do not have dominant variance in one of the principal directions. Fortunately, such signatures are the exception rather than the rule as people tend to sign in a predominantly left-right fashion.

[37, 36] transform a signature into canonical form in the frequency domain. The first derivative of the sampled coordinates are obtained to reduce end-point distortions. This derived signal is converted to the frequency domain by applying the Fourier transform. Transformations are carried out in the frequency domain to achieve rotational and scaling invariance. It is done this way as the intended operations are conceptually much simpler in this domain. Smoothing is achieved by zeroing small amplitude frequencies. After the transformations, the signal is converted back into the time domain for modelling.

The neural approaches used in [41, 18] call for a fixed sequence length. This is achieved by linear-time normalization of a signature's spatial time function  $(x(t), y(t))$ . The data is resampled with respect to the time parameter. This might inherently distort the input signature [68], especially if the resampled sequence length is shorter than the original. They exclude important information carried in frequency bands excluded by the resampling<sup>6</sup>. Other modelling methods such as dynamic time warping do not require sequences of a fixed length. For these methods however, computed distance values between two sequences are often later subjected to length normalization.

Effective preprocessing is unavoidable if a verification system is to attain commercially acceptable performance and work for a wide variety of hardware. For credit card transactions, [51] regards a 1% false acceptance rate and a 7% false rejection rate as reasonable. Given that there is currently hardly any verification done and the potential user resistance to having one out of every 14 signatures rejected, we would rather see these numbers reversed. [40] requires a base-line performance of 0.05% FRR and 20% FAR for inclusion of the results in their survey. In practice though, the required error rates depend largely on the penalties incurred

---

<sup>5</sup>pen tilt is a measurement of the angle the pen makes with the surface of the tablet

<sup>6</sup>commonly known as the Nyquist frequency

by making an error of each of the two types in the specific scenario [55]. Reference [54] underlines the need for preprocessing. They perform very little preprocessing and conclude that in order for their modelling approach to obtain acceptable error rates, more attention needs to be paid to preprocessing. If used in a sensible fashion, the information removed during a normalization phase can, when isolated, be used to improve a system's performance [42]. Furthermore, depending on the modelling approach, normalization might not be necessary as far as rotation is concerned. In such cases, only rotational invariant features are used to represent a signature [41]. The absolute velocity of the pen tip is one such feature.

## 2.5 Signature Modelling

The modelling technique is employed at the heart of an ASV system. Even though the performance of a system depends largely on the degree to which all the aspects of the system work together, the applicability of the modelling technique and ability to recognize genuine signers *and* forgers are the most important factors in the quality of a verification system. It is therefore no surprise that it is in this part of the field where the most effort is exerted.

This section summarizes various modelling approaches applied to ASV. The list is by no means complete but we hope it covers most of the major research current directions in the field.

### 2.5.1 Dynamic Time Warping

Dynamic time warping (DTW) stems from the field of dynamic programming. The general idea of dynamic programming is to find a least cost path through a cost matrix in an attempt to optimize some process. The challenge is to define a suitable cost function for the problem at hand. Dynamic time warping finds a non-linear time alignment between two sequences to compensate for non-regular stretching or compression in the sequences. If two sequences show similar overall shape, DTW can find a unifying time function which will align the two sequences in a way minimizing the distance between them. If they are not of similar shape, DTW will find some alignment but the warped sequences will remain far apart. It goes about

finding an alignment by placing one of the sequences on the vertical axis of a discrete matrix and the other on the horizontal axis. Each matrix position is then set to the cost of aligning the partial sequences up to that position on each sequence so the distance between the sequences are minimized. Various cost functions can be used each having particular characteristics. For an in-depth discussion of DTW, see [57, 58].

One of the earliest studies on ASV [72], uses dynamic time warping to find a non-linear alignment between signatures. Prior to alignment, the equivalent force function varying over time is calculated from the dynamically sampled data. The function is computed as follows

$$f(t) = d'' + [1 + \frac{\mu P(t)}{v}]d' \quad \text{with } v = \sqrt{x'^2 + y'^2} \quad P = \frac{p}{M}$$

where  $d$  is the displacement function,  $p$  is the writing pressure function,  $v$  is the writing speed,  $1$ ,  $\mu$  and  $M$  are the viscosity coefficient of the human hand, the friction coefficient between pencil point and writing surface and equivalent mass hand-pen coupling respectively. This function is supposed to be a more direct representation of the control timing information of the writing movement than for instance the pencil point displacement. The force functions of two signatures are aligned by DTW. Extreme time warping during alignment is prevented by placing constraints on the DTW routine. The output of the alignment is a normalized distance between the two signature representations. This distance is compared to a personalized threshold which is determined experimentally. Most verification systems compute some measure of similarity or distance between two signatures and compare this to a threshold value which in itself can be obtained in various ways.

It might be attractive to reduce the task of signature verification to two steps. That of low-level feature detection followed by some standard method of feature-vector comparison. The price one pays for this simplification is that the overall result is only as good as the features selected. The method described in Section [32] is based on the approach of representing signature data by functions of time (instead of a number of low-dimensional parameters or features). The study makes use of dynamic time warping and geometric shape analysis to perform verification. The DTW is used to match the speed signals of two signatures. It is believed that DTW should not be used to compensate for other variations such as Euclidean shape transformations which will be the case, should positional functions be aligned without

being normalized first. Time warping is necessary as writing speed will change from one signature to the next, as will its consistency over different regions. The slant might increase as the writing speed increases resulting in non-regular deformation of the speed signal. Non-linear time alignment produced by DTW, for instance, can compensate for this. Apart from alignment distances inspected for verification, further checking is done by comparison of signature segments. A segmentation of a signature into pieces which exhibit little oscillations in its various features can be achieved by using points of high curvature as the segmentation boundaries. The curvature signal has to be computed from second order derivatives though, which are numerically unstable. Therefore, the simple observation that points of high curvature coincide with points of low speed, is used instead to obtain a segmentation of a reference signature. The reference signature is selected as the signature in the training set which deviates the least from the other signatures during DTW. A segmentation is created by dropping pieces of the signal where the speed component is less than a threshold percentage (e.g. 15%) of the mean speed. For each of these segments, a template average shape is estimated up to an affine transform which allows for differences in location, scaling, orientation and shear. This template is described as

$$Y(u) = \begin{bmatrix} x(u) \\ y(u) \end{bmatrix} = A(u)F(u) + \mu(u) + e(u), 0 \leq u \leq 1$$

where

- $F(\cdot)$  is an idealized template or "mean" signature for the writer
- $A(\cdot)$  is a  $2 \times 2$  affine transformation matrix
- $\mu(\cdot)$  is an affine transformation vector
- $e(\cdot)$  is a stochastic vector of departures from the model

The estimation procedure discovers the parameters for these elements. These average segments are then concatenated to form a template signature. Verification is carried out at both the DTW stage and the affine transform stage. If the discrepancy at the DTW stage is not big enough to conclude the authenticity, the signature is segmented and the segments affinely



transformed to match the template. The least squares distance between the template and suspect signature segments are then used to decide on the authenticity. This phase inspects a possible forgery for unacceptable shape variations. It is extremely unlikely that a forger can mimic both the shape and relative speed with which a person signs their name. This is even more so if the signature to be forged contains exotic flourishes which are often illegible but consistent in the victim's signature. As can be seen, [32] performs checks on both the speed and shape of a signature.

In [50], signature dynamics are obtained from cameras. A normal pen is tracked by applying optimal signal detection techniques to images sampled from a digital camera. The tracking algorithm cannot distinguish between the up and down state of the pen so the entire pen trajectory is recorded. Various different parametrizations of the signatures were tested in this study. The affine arc-length parametrization has been found to be superior to arc-length and time parametrization. The sequences are aligned by dynamic time warping even though the parametrization is not necessarily by time. A reference signature is obtained by finding the training signature which shows the least deformation during alignment with all the other samples. The average of this alignment with all the other signature samples represents a prototype signature in the system. The distance between a reference signature and a test signature is evaluated in various different ways in search for the optimal distance measure for this system. A harmonic mean measure was found to outperform residual distance, correlation and weighted correlation when establishing time correspondence between two curves. The study claims, somewhat contradictory to the general opinion, that dynamic information is of less importance than static information during verification.

The feature based approach described in Section 2.5.5 is augmented by stroke-direction coding (SDC). With SDC, [37] attempts to model hand movements that produce a signature. A signature is divided into a fixed number of time-ordered links called strokes, where each link is approximately of the same length. A stroke is described by a number indicating the general direction of pen movement within the stroke. A non-linear alignment through dynamic time warping is used to establish the deviation of the SDC vector of a test signature from a reference SDC vector. This deviation and the feature based error measure are combined for verification.

DTW is quite often used to find an alignment between sequences. Other approaches do exist

however. In [69], a signature is represented by a static feature sequence which is the sampled  $(x, y)$  sequence and a dynamic feature sequence which is the velocity computed from the static sequence. To match an input signature with a reference signature, the two sequences have to be aligned. This study proposes a technique called split-and-merge. In contrast with dynamic time warping which is a piecewise advancing match algorithm, split-and-merge is a top-down approach. It proceeds in a recursive fashion by splitting the reference sequence in the middle and the test sequence at such a place that after refining the two subsequences and merging them, it best matches the reference sequence. A subsequence is refined by removing the non-uniform compression or spreading among sub-patterns relative to the reference sequence. The refined subsequences are merged and interpolation is used to make the reference and test sequence the same length. After this, the distance between two sequences are measured and compared to a threshold value derived from the training set. An input signature is deemed genuine if both of its coordinate and velocity distance from the reference template are less than the respective coordinate and velocity thresholds. Results show that there is a split-and-merge recursion depth beyond which no performance gain is achieved.

We now briefly summarize the results of a study which compares DTW with other approaches. [53] discusses Dynamic Time Warping, Regional Correlation and Skeletal Tree Matching. The idea of regional correlation is to cut signals into regions and to correlate corresponding regions over different time lags to find the best possible match. The dynamic time warping variation used in this study is based largely on work done in the field of speech recognition. For skeletal tree matching, a tree representation is created for each of the two signals being compared. The tree representation seeks to capture peaks and valleys in the waveform together with their self-embedded structure. The methods are compared with respect to verification error rates, execution time and number and sensitivity of parameters. The comparisons are extended beyond normal signatures to handwritten passwords and initials. Furthermore, the tests are conducted using positional, velocity and acceleration signal representations, respectively. A variance analysis on the individual results shows that no algorithm consistently outperforms the other.

## 2.5.2 Hidden Markov Models

Apart from automatic signature verification, hidden Markov models (HMM) are also used with a great deal of success in automatic speech recognition and molecular biology. Essentially, they extend the well-known concept of Markov chains and are thus founded on solid statistical principals. HMMs comprise of a state graph connected by probabilistic transitions. Each state can accept an observation with some probability. The observation at each time instance need not be single-variate and can be either discrete or continuous. HMMs allow for the modelling of non-linear time variance in sequences of observations by dictating transition probabilities between states or imposing explicit state durations. This proves to be a handy feature when working with signatures which exhibit time warping amongst different samples originating from a single signer (see Section 2.5.1). Such a time-warping profile can serve as a distinguishing feature if captured by a model which is indeed the case for HMMs. Generally speaking, verification systems based upon HMMs are concerned with finding appropriate sequences of observations to represent a signature and to attach sensible semantics to model states. For a more in-depth discussion of HMMs, see Chapter 3 and references [56, 57].

The absolute angular direction of signature samples as a function of the distance along the signature trajectory is used to represent a sampled signature in [71]. This sequence of angles are divided into a fixed number of segments. A formula incorporating all the angles in a segment is used to calculate a discretization code representing a segment. This sequence of codes is then presented to a HMM. The theory provides for the calculation of a likelihood that a sequence was generated by the process being modeled by the HMM. This value for a test signature is compared against a threshold likelihood value to verify the authenticity of the signature. This approach in a sense counteracts the time warping ability of HMMs by implicitly assuming that an equal segmentation will group similar subparts of a writer's signature. The equal segmentation adopted from speech recognition cannot be applied with equal success to ASV due to the huge difference in the amount of samples available. [51] agrees with our view as signature sequences are not long enough for a model to recover from segmentation errors.

In [23], samples are blocked into segments bounded by points where the velocity  $v_y$  in the

$y$  direction crosses zero. It is argued that segmenting on these points results in a size independent representation. A 32-component feature vector is derived for each segment. Linear discriminant analysis is performed on this feature vector and the  $N$  most discriminative features are selected to represent segments. A left-right hidden Markov model is used to model the sequence of feature vectors. An adaptive threshold for a signer is computed from the average likelihoods for the training set combined with a system dependent offset.

Much the same as in [37], [36] reports on a method combining global and local features. For a description of the global features, see the survey in Section 2.5.5. For the local feature-based part, a hidden Markov model with explicit duration modelling is used. This results in a variable duration hidden Markov model. This model is also referred to as a hidden semi-Markov model (HSMM). In [36], a specific HMM configuration is used to approximate a HSMM. Each HSMM state is decomposed into a number of unit-duration substates, resulting in a HMM with a larger number of states than the HSMM. Each sample in a signature is represented by an inclination angle and the difference between adjacent inclination angles. These values are quantized for use with a discrete HMM. In such a HMM, no assumption about the distribution of the data needs to be made (as opposed to continuous HMMs). The calculated likelihoods are divided by the number of sample points to reduce the effect of signing time variations on the algorithm. The difference between the likelihood for a test signature and the average likelihood for the training set is used as an error measure to determine the authenticity of a signature. The global and local errors are combined using a Euclidean distance measure to reach a conclusion. Results show that the combined use of global and local features perform better than any of the two parts on their own.

[48] extends the idea of signature verification to a system where a signature is substituted by a written password. This means that not only does a forger have to imitate the writing dynamics, but also guess the statics, i.e. the password. A written sequence is normalized to a horizontal baseline. It is then segmented into strokes delimited by consecutive minima of the absolute pen-tip velocity. Each stroke's net direction is obtained by placing the starting point of the stroke at the origin of the Cartesian plane and observing the quadrant of the end-point of the segment. This results in a discretization of size four. An element for recording pen-up events is added giving a codebook size of five elements. The sequence of discretized

observations is then modified by repeating symbols proportional to the length of a segment. This modification enables the algorithm to make better use of the time warping ability of the HMM. A very compact HMM with only five states is used to model the sequence of symbols.

### 2.5.3 Filters and Frequency Domain

The Fourier transform is probably the most widely used mathematical tool in signal processing applications today. It has found its way into signature verification as well. This section explores studies using what we deem to be more traditional signal processing techniques including the Fourier transform and spectral analysis made possible by it.

Different signature samples of a writer almost always exhibit instabilities of some kind. [64] introduces a distortion measure to deal with this fact. This distortion measure is based on DTW and serves as a first step in the verification process. If this phase cannot decide conclusively on the authenticity of a suspect signature, a next phase based on spectral correlation is employed. For this, preprocessing of a signature consists of resampling a linear interpolation of the signal and including velocity information in the new signal. This signal is transformed into the frequency domain by a FFT. Linear correlation is used to find the similarity between the spectra of an input and reference signature. As usual, the correlation coefficient is compared to a threshold value. In calculating the correlation coefficient, the weight of each frequency component depends on the stability of the component as deduced from the training set.

In [67], the sampled  $(x, y)$  coordinates of a signature are converted to the frequency domain by the fast Fourier transform. To smooth out sharp spikes in this frequency spectrum, the log of the Fourier coefficients are taken to represent the signature as a logarithmic spectrum. Through principal component analysis based on scatter matrices, only a small amount of these coefficients are extracted to represent a signature. The similarity of the changing rate of coordinates between two signatures can be characterized by the similarity of the coefficients of logarithmic spectrum. A reference template for a signer is obtained by taking the mean values of the transformed training sequences.

[22] uses a local verification strategy based on spectral analysis performed on fundamental

components. Components are defined to be pieces of writing included between a pen-down movement and the successive pen-up movement (called pen-down singularity as opposed to pen-up singularities). It is claimed that these singularities can occur only in positions which are rather constant in the signatures of an individual. Stability in the positions of singularities allows identification of the finite set of fundamental components of each signer. The existence of a finite set of fundamental components in the signature of an individual makes forgery detection by a component-oriented verification system possible. During enrollment, a knowledge-base for an individual is created containing a component reference table and a structural description graph. The component reference table contains the features representative of the classes of fundamental components of a signer. The structural description graph reports the acceptable sequences of fundamental components in the genuine signatures. For each component, a 5-dimensional topological feature vector is created to describe the component. A k-means clustering technique is then used in three phases to detect different component clusters. These phases are described as Initial Clusters Recognition, Clusters Growing and Final Clustering. The study finds that small variation from these clusters confirm the stability of these topological features in the writing process and their effectiveness for the clustering of the fundamental components. The classification of each component of the reference signature permits the identification of the sequences of fundamental components. The algorithm creates a graph allowing for the different component sequences as they occur within the training set. Many differences may exist among the components within each cluster such as subtle shape variations or dynamics. To detect such differences among components belonging to the same class, a subclustering procedure using particular Fourier descriptors is used. Only the first few Fourier descriptors are used due to the band-limited nature of the signals produced by the human writing system. A maximum distance algorithm is used to split clusters into subclusters based on the differences in Fourier descriptors. Verification is done in a two-step fashion. The first step dictates that for a suspect signature to be classified as authentic, its sequence of components must match a possible sequence in the structural description graph of the claimed signer. If this step is successfully completed, the second step verification is performed where each cluster is verified individually. The Fourier descriptors are used in a distance measure against a threshold value. If any component fails the test, the signature is classified a forgery. The threshold value for each cluster is automatically derived

using the worst verification result obtained from the genuine components.

Velocity signals can be derived from positional signals. For the velocity signals  $v_x$  and  $v_y$ , the autocorrelation functions  $R_{v_x}$  and  $R_{v_y}$  are calculated. These signals are then regarded as the input and output, respectively, of a finite impulse response (FIR) filter in [45]. The impulse response is obtained by minimizing the least-square error between the autocorrelation signals. A reference vector of impulse responses is calculated from random samples from the training set. The distance between the impulse response of a suspect signature and the reference impulse response is compared to a threshold value to decide on the authenticity.

#### 2.5.4 Artificial Neural Networks

Artificial neural networks (ANN) are used today in a wide variety of applications. Some of these include stockmarket prediction, medical diagnosis, seismic event prediction, speech recognition and artificial vision to name but a few. ANNs are an active research field and automatic signature verification is no exception. For a gentle introduction to various different neural network architectures see [43].

The linear predictor coefficients (LPC) cepstrum is defined as the Fourier representation of the logarithmic amplitude spectrum of a signal. In [68], cepstral coefficients derived from LPCs of the writing trajectories are calculated as the features of signatures. These coefficients are fed into a multi-layer perceptron (MLP) with multiple input nodes and a single output node. The MLP is selectively trained with back-propagation training meaning the weights are not updated if the desired output is closer than a certain predefined value from the network output. For authentic signatures, the desired output is set to one and for forgeries, it is set to zero. During verification, the LPC cepstrum features of a signature are presented to the trained network and if the output is larger than a threshold value (e.g. 0.5) the signature is accepted as authentic, otherwise it is rejected. A potential problem with this system is the need for negative examples i.e. forgeries. These would be difficult to obtain for a large scale production system and the use of random forgeries might result in less than optimal performance.

ANNs can learn from training examples and have the ability to compress information. Compression is an important consideration in [18] as an 80 byte restriction is imposed on the study by the fact that the model needs to be stored on a credit card magnetic strip. A signature is resampled to a fixed number of points by interpolation. Two such resampled signatures are then presented to two subnetworks based on the time delay neural network paradigm. The two subnetworks are joined at the output layer and the objective is to minimize the cosine distance of two feature vectors extracted by the subnetworks. The cosine distance is calculated as

$$\frac{f_1 \cdot f_2}{|f_1||f_2|}$$

Pairs of input are presented to the network. For pairs of genuine signatures, the desired cosine distance are desired to be 1.0 and for genuine-forged pairs -1.0. Once the network is trained it can be used for verification by presenting training signatures to one of the subnetworks and assuming the output of the network to be a multivariate normal distributed feature. The decision process then becomes a task of inspecting the likelihood value from such a density function.

A time delay neural network (TDNN) is an extension to the basic MLP. Tap-delay lines are added on the input layer to facilitate sequences of data rather than static patterns as is the case for the MLP. A signature is modeled by a TDNN in [59]. Feature signals such as velocity, direction and curvature of the pen trajectory are added to the sampled signals. For a specific signer, a TDNN is trained by creating a network with default structure and input window size and applying the error backpropagation learning algorithm. Exemplars are presented in an iterative fashion. Regulated structural changes are imposed and network input window sizes changed according to a specific strategy until the network error ceases to decrease.

A syntactic neural net is a connectionist architecture with the ability to infer grammars from training patterns. A strictly hierarchical context-free grammar is defined in [44] to be inferred by such a network. A signature's positional  $(x, y)$  information is sampled from a tablet over time. The samples are quantized into an alphabet of eight direction vectors and a null vector for no movement. A non-temporal connectionist parser (NCP) is then used for learning and



verification. In theory, the NCP learning and parsing time scale linearly with the pattern length.

Different neural architectures are compared in [41]. A signature is normalized by resampling from a linear interpolation to obtain a sequence of a predefined fixed length. The absolute velocity is used as it is shift, rotation and translation invariant. It is related to  $(x(t), y(t))$  as

$$|v(t)| = \sqrt{\Delta x(t)^2 + \Delta y(t)^2}$$

Three different neural architectures are tested: TDNN - time-delay neural network, IONN - input-output neural network and BMP - Bayes multilayer perceptron. These methods appeal to ASV since they act as single systems which automatically extract discriminant features and execute optimal classification in the sense of the Bayes decision rule. Only skilled forgeries are employed in the experiment as it is argued that the real nature of the forgery space is unknown and testing results for random forgeries hardly provides a high degree of reliability and robustness of a ASV system. The performance results reported in the study reveal that it is essential to have forgery training data for NN training. Results show that the BMP outperforms the other architectures suggesting it explores global features whereas the other explore local features. The sequence used in this study is fairly long making it difficult for TDNN and IONN to effectively discover discriminating evidence in local features if the dimension of the data is not high enough as is the case here.

The ART1 neural network is used in [62] to do signature verification. The pressure pattern sampled from a digitizing tablet is quantized into a binary string of fixed length. A reference pattern is obtained by using the mean pattern for the training set. A vigilance parameter for the ART1 network is derived by inspecting the similarity of the reference pattern to the training patterns. The ART1 network is then trained in the normal sense. Verification is done by presenting a quantized pressure pattern under suspicion to the input nodes and comparing the output to the vigilance parameter to reach a verdict on the authenticity of a signature. The study states that the intended use is for a first stage screening only in a verification system. This scheme is applied to Chinese signature verification where there is generally more pen-up/down transitions than in other languages. which makes this approach viable.

As can be seen from the mentioned studies, a common problem is the need for negative examples (meaning forgeries) when training neural networks. ANNs function by positioning decision surfaces between classes of data rather than positioning model parameters on the data as is the case with for instance HMMs. This problem can be bridged by applying random affine transformations to authentic signatures within an acceptable threshold to fabricate forgeries. It remains to be explored though how effective this approach will be compared to using real forgeries.

### 2.5.5 Statistical Modelling

The well established field of modern statistics provides a solid basis from which to build pattern recognition systems. Various statistical techniques have been applied to ASV. In a sense, one can argue that most ASV systems will incorporate some fundamental statistical concept somewhere. This section contains studies which makes use of predominantly statistical concepts.

Feature-based statistical methods apply transformations to the data which result in a set of features. They are chosen to expose differences between genuine signatures and forgeries. The feature extraction process can be seen as signature compression. The challenge is to extract features which do not discard relevant information. Dynamic features describe aspects which are not apparent from an examination of a copy of a signature. A forger needs to duplicate the shape and the way it was signed. Therefore the verification procedure must include a mixture of both shape and dynamic-related features. [52] use a set of 25 features. Some examples are the total signature time, the root mean square speed, the integrated absolute centripetal acceleration, a direction histogram ( $0-2\pi$  divided into eight sectors) and the X,Y speed correlation. It is desirable for shape-related features not to be strongly correlated to dynamic features. There may be the extra constraint that the parameters of features must not exceed the storage limit for a particular application (e.g. 80 bytes for credit cards). A feature is a good discriminator between genuine and forged signatures, if its values on genuine signatures constitute a cluster which can be separated with high accuracy from that of forgeries. To verify a signature, its feature vector is computed and compared to a template vector by some distance metric. The study reports on Euclidean, Mahalanobis and Quadratic

distance models. The study assumes the feature vectors to come from mixtures of multivariate Gaussian probability density functions. It is claimed that the statistical properties of genuine signatures should be reasonably predictable as they are produced by a single known signer (in contrast with forgeries for which no a priori knowledge is available). Decision rules are defined for both the cases where forgeries are and are not available.

In [37], 23 global features are used. These features are divided into roughly two categories: shape-related and dynamical features. Care is taken to ensure that the shape-related features are not strongly correlated to the dynamical features. For each of the features, the mean  $\mu$  and the standard deviation  $\sigma$  are calculated from training samples for a specific signer. These are then used in a joint distance measure to determine the degree of similarity of an unknown signature. This study shows that a verification system need not comprise of only a single modelling approach. The feature-based model is further augmented in [37] by what the authors call stroke direction coding as described in Section 2.5.1. The results show that the combination of SDC and this feature based approach outperforms each approach on their own.

One of the most attractive qualities of feature based verification systems, is the relatively small amount of memory needed to store a signature model. This is an important consideration for many current commercial applications where storage ability is restricted e.g. credit/SMART cards.

A signature can also be seen as a stochastic process. In [46], it is shown how the random impulse response for a system is calculated where the relationship with the sampled  $(x, y)$  signal is

$$y(t) = \int_0^T h(t, \tau)x(\tau)d\tau$$

with  $h(t, \tau)$  the random impulse response. For verification, a distance measure between two sequences of random impulse response parameters is defined.

[47] approximates a signature by a piecewise linear function i.e. the locus of pen movement is approximated by line segments. Each line segment is depicted in magnitude/argument form as is commonly used to represent complex vectors. For this sequence of magnitude/argument

pairs, a two-dimensional AR model is defined and its parameters are obtained by solving a set of simultaneous equations. The cross spectral density is calculated for the AR parameters. Then the discrete cosine transform is performed on the cross spectral density and the logarithm of the transform coefficients is the features representing a signature. The distance between the test signature's feature vector and a reference feature vector is calculated and if within an acceptable threshold difference, the signature is classified as authentic. The reference feature vector is built from a randomly selected subset of the signature samples of a subject.

A vector autoregressive (VAR) model is explored in [54]. A sampled  $(x, y)$  sequence is resampled to a fixed sequence of length 512. This sequence is divided into a fixed number of sections. The sections are then each modeled by a VAR. It is argued that the VAR coefficient matrix eigenvalues, the scalar VAR coefficients, the mean vectors and the noise measures built into the model, own intraclass invariant properties. This makes them excellent candidates as features for classification and verification. The eigenvalues of the VAR model coefficient matrices are used instead of the matrix elements themselves to reduce the feature vector size. The distance measure used to compare a suspect feature set with a reference feature set involves a discretization of the features to obtain likelihood values from a frequency matrix. As usual, a threshold distance decides on the authenticity of a signature. This study also compares this approach to a subset one-dimensional approach to assess whether the extra parameters obtained in this study yields a significant performance increase. The study concludes that, even though the results have improved, the improvements are not statistically significant to warrant the computational overhead.

### 2.5.6 Optimal Feature Selection

As we have seen in the previous section, feature-based systems compute features from sampled signatures where each feature represents some characteristic of a signature. This is called feature extraction (see Section 2.5.5). Features can be chosen with the hope that they constitute a concise representation of a signature. Considerations taken into account when selecting features in [52] are

- they must be insensitive to variations in genuine signatures
- they must be good discriminators between genuine signatures and forgeries

Because no *a priori* knowledge is available about which of the vast array of possible features will give the best discriminating power, a feature set might contain a lot of redundant information with no guided way of pruning them. The objective of feature selection (as opposed to feature extraction) is to obtain a reduced set of features which contains essentially all the discriminating power of the original set. Feature selection addresses the following aspects of a feature based verification system:

- efficiency through the removal of redundant information
- speed by reducing the dimension of the feature vector
- performance by working only with an optimal feature set

In general, most feature selection techniques follow the same basic procedure. The starting point is a large set of features which the analyst believes to be useful for discriminating between samples. The discriminating power of each of the features or combinations of features is determined by performing statistical tests on a training set of data which is believed to adequately represent the population. The combination of features which yields the best performance (by some criteria) and which contains the minimum number of features is deemed the best feature set. Furthermore, the optimal feature set need not be the same among different signers. Different approaches to finding an optimal set are reported.

[42] defines a set of 49 normalized indicators extracted from a positional signature signal sampled from a tablet. For a subject, the  $k$  most important features are selected amongst these by ordering the features according to their maximum distance from the rest of the entire population. The distance measure involves the mean and variance of a feature obtained from a training set. This results in an optimum individualized feature set. The study also presents a common feature set composed of those features with the highest frequency of appearance in all the individualized feature sets.

The simplest methods select features through trial and error or brute-force [19]. Such an approach is time-consuming as there can potentially be a vast number of combinations to search through. Sub-optimal searches reduce the size of the search space by imposing certain structural or traversal restrictions on the search tree. Parallel strategies for feature vector construction is considered in [27]. It is shown that there exist inherent parallelism in the feature selection process which can be used to perform the task on a parallel computer. Various parallel algorithms are implemented and compared. The possibility of using a transputer is attractive as it reduces the amount of time needed to select an optimal feature set.

As the name would suggest, the genetic algorithm finds its origins in the field of Biology. It is based on the way living organisms evolve on a genetic level to attain the best genes suitable for their situation. The algorithm employs these principles to find an optimal solution to a problem at hand. The challenge here is to find an encoding for the problem in terms of chromosomes. [70] shows how this algorithm can be applied to the problem of feature selection. It is necessary to select features of signatures which can overcome the dilemma of intra- and inter-personal variability. [70] states that not all sampled points of a signature are necessary for verification. Experts concentrate on some particular parts which have distinguishing features when they engage in signature verification. What is more, different features in different parts of the signature must be used for the verification. A signature is a sequence of time ordered data and the combinations of features are unlimited. It is very difficult to predetermine an optimal set of features. The result of the selection is not even unique. The genetic algorithm has a high degree of ability to solve this problem. This study presents a novel method to select partial curves and features of the curves of signatures for verification using the genetic algorithmic. The study also proposes a new crossover method in order to determine the number of partial curves. The described system consists of a feature selection part and a signature verification part. The location of partial curves and the features of the curves used for the verification are encoded into the chromosome. The length of a chromosome i.e. the number of loci, corresponds to the number of partial curves of the signature. The genotypes are then modified by the genetic algorithm using the local improvement mechanism. Each chromosome is evaluated by a fuzzy network and the chromosome's fitness value is calculated. The one with the highest fitness value is selected. This elite chromosome includes the best set of partial curves and features of each curve for a true signature.

It is perhaps suitable to end the modelling section with a paper which dares to challenge accepted beliefs. In [51], the author disagrees with the general notion that velocities and forces plays a pivotal role in ASV. The reason for this is that no evidence could be gathered to show a signer's pen dynamics are consistent enough to be used as distinguishing features in verification. Foremost, for two signatures to be declared as produced by the same individual, it is necessary for the shape of both to match closely. The author perseveres that for ages we have relied on visual examination of signatures to decide authenticity. He finds it difficult to justify the jump to time related information. The author claims that all the subjects in his study could produce their signatures both as a reflex and deliberately without visual deterioration. The study presents a number of novel aspects to ASV. The concept of *jitter* is introduced as a quantity measuring the act of a forger constantly correcting the pen trajectory to conform to an *a priori* curve. To make a signature independent of orientation and aspect, it is normalized. This is done by fitting a polygon to the ordered set of samples and use the global axes of maximum and minimum inertia running through the global center of mass and rotate the signature to normalize these axes. The rotated signal is then scaled to normalize the aspect. This normalized signal is then parametrized over its length (instead of time). By using a moving coordinate frame, the center of mass, torque and moments of inertia at the center of the window is calculated using a Gaussian weighting function. These derived signals are used to characterize a signature. To compare a signature to a reference characteristic function set, the two sets of functions are *length* warped (in contrast with the more familiar time warping) as to maximize the sum of the weighted cross correlation of each function with respect to its model. The error between each characteristic function and its reference model is computed. The study then uses what it calls the harmonic mean to quantize the global error based on the joint error for the jitter, aspect and warping distance. The study describes in detail various databases used for tests and presents a real implementation combining SMART card technology, a proprietary digitizer and a notebook computer. This study highlights various topics which are central to the problem of ASV.

## 2.6 Conclusion

Signature verification research has found its way into a number of commercial applications. A simple websearch reveals various commercial ventures which utilize automatic signature verification. Applications range from financial transaction authentication to restricted area access control. Only time will show though if signature verification can hold up to more hyped biometrical authentication schemes such as fingerprint scanning. The absence of a representative test database to the research community could prove to be a deciding factor in the future widespread use of ASV.



## Chapter 3

# Hidden Markov Models for Sequence Processing

The theory of hidden Markov models (HMMs) was first introduced in a series of papers by Baum and colleagues in the late 1960's [7, 9, 8, 11, 10]. Since then, it has found its way into many research areas most notably speech recognition [56] and molecular biology [39].

As the work described in this thesis uses hidden Markov models extensively, we have included a chapter on the topic. It should be noted however that this chapter will not attempt to replace the excellent introductory work on HMMs found in papers such as the seminal tutorial by Rabiner [56].

### 3.1 Introduction

Hidden Markov models extend statistical models known as Markov chains. Markov chains arise naturally in biology, psychology, economics and many other sciences. We shall proceed with an overview of Markov chains and then extend the concept to hidden Markov models.

## 3.2 Markov chains

Real-world processes generally produce observable outputs which can be characterized as signals of either discrete (e.g. weather classified into sunny, cloudy and rainy) or continuous nature (e.g. features extracted from speech signals). The non-deterministic fluctuation of the weather state is an example of a system which may be expressed (i.e. modelled) by a Markov chain. Such a model can then be used to predict the likelihood of a certain state some time in the future.

More precisely, Markov chains are used to model phenomena exhibiting a sequence (i.e. chain) of fixed length periods during which any one of a set of  $N$  distinct states,  $S = \{S_1, S_2, \dots, S_N\}$  can be assumed. Transitions between states occur over time and are expressed by probabilistic means. A matrix of transition probabilities links states by giving the probability of being in a state for the next time period, given the current state of the system

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i), \text{ for } 1 \leq i, j \leq N.$$

The entries  $a_{ij}$  need not necessarily be non-zero for all  $i \times j$ ; if they are, then the model is said to be a fully connected model or *ergodic* model. The assumption that the state at a certain time is dependent only on the previous state, is called the first order Markov assumption. This need not necessarily be the case. If  $n$  is the length of the state history influencing the choice of the next state, the model is said to be an  $n^{\text{th}}$  order Markov model but, as is most often the case,  $n = 1$  implying first order Markov models. The Markov assumption simplifies matters significantly, however, for many complex processes the first order assumption may lead to a less than accurate expression by the model. Nevertheless, since such simplified systems may often be more readily subjected to analysis, we bring ourselves to live with the shortages, bearing in mind the possible inaccuracy of the results. We may perhaps compensate for them in other ways through domain specific knowledge in order to tap from the sound formalism of Markov models and especially hidden Markov models. Recently, it has been shown how efficient higher order hidden Markov models can be realized [24].

Figure 3.1 depicts all possible first order transitions between our chosen weather states. For

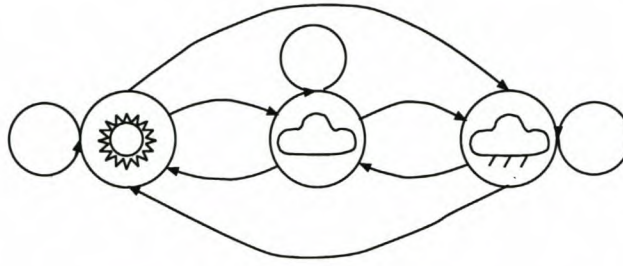


Figure 3.1: **Weather state transitions:** A fully connected (ergodic) model configuration.

$N$  distinct states, there are  $N^2$  transition probabilities which, as previously stated, can be collected into a state transition matrix  $A$ . For the weather example with  $S = \{S_1 = \text{sunny}, S_2 = \text{cloudy}, S_3 = \text{rainy}\}$ , the transition probability matrix becomes

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Entry  $a_{ij}$ , with  $i$  indicating the row and  $j$  indicating the column, is the probability of making the transition from state  $i$  to state  $j$ . Thus,  $a_{22}$  is the probability of the weather remaining cloudy given that it was cloudy for the previous time period. Because  $a_{ij}$  is interpreted as a probability, the row entries must adhere to stochastic constraints with

$$a_{ij} \geq 0, \text{ for } 1 \leq i, j \leq N, \text{ and}$$

$$\sum_{j=1}^N a_{ij} = 1, \text{ for } 1 \leq i \leq N$$

The probabilities remain stationary over time which often proves to be an unrealistic assumption.

There is still one missing part of information in defining the weather Markov model. The vector  $\Pi = \begin{pmatrix} \pi_1 & \pi_2 & \pi_3 \end{pmatrix}$  denotes what the probable state of the weather was at time  $t_1$ . We have now fully defined a first order Markov model  $M = \{S, \Pi, A\}$  with

- $S$  : states,
- $\Pi$  : starting state probabilities and
- $A$  : transition probabilities.

Such a system is called a *Markov process*.

We could now ask questions regarding the model for instance: What is the probability that the observation sequence  $O = \{S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$  was generated by the model  $M$ ? That is, what would the model say is the chance of having the weather start out as rainy and then be rainy-rainy-sunny-sunny-rainy-cloudy-rainy. This can be expressed as

$$\begin{aligned} P(O|M) &= P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|M) \\ &= P(S_3) \cdot P(S_3|S_3) \cdot P(S_3|S_3) \cdot P(S_1|S_3) \cdot P(S_1|S_1) \cdot P(S_3|S_1) \cdot P(S_2|S_3) \cdot P(S_3|S_2) \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \end{aligned}$$

Another question which leads to the notion of a probability density function is: Given the model is in a known state, what is the probability that it would stay in that state for *exactly*  $d$  days? The observation sequence would be

$$O = \{S_i^1, S_i^2, S_i^3, \dots, S_i^d, S_{j \neq i}^{d+1}\}$$

where the superscripts merely indicate the time instance. We express the probability as

$$P(O|M, q_1 = S_i) = 1 \cdot (a_{ii})^{d-1} (1 - a_{ii}) = p_i(d)$$

with the 1 referring to the certainty of our knowledge about what the state at  $t_1$  is and  $(1 - a_{ii})$  referring to the mandatory transition out of state  $i$ . We call this quantity  $p_i(d)$  the discrete probability density function of duration  $d$  in state  $i$ . Using  $p_i(d)$  we can now calculate the expected duration of remaining in state  $i$ , denoted by  $\bar{d}_i$ , given the system started in state  $i$  by

$$\begin{aligned} \bar{d}_i &= \sum_{d=1}^{\infty} d p_i(d) \\ &= \sum_{d=1}^{\infty} d (a_{ii})^{d-1} (1 - a_{ii}) \\ &= (1 - a_{ii}) \sum_{d=1}^{\infty} d (a_{ii})^{d-1} \\ &= (1 - a_{ii}) \frac{1}{(1 - a_{ii})^2} \\ &= \frac{1}{1 - a_{ii}} \end{aligned}$$

using the well-known infinite series identity

$$\sum_{n=1}^{\infty} nx^{n-1} = \frac{1}{(1-x)^2}, \text{ for } |x| < 1.$$

The discussed model is called an *observable* Markov model since the output of the process is a set of observations at each instance of time where each model state corresponds to an observable event. There are, however, instances where this modeling technique is too limited to model the process under inspection. This leads us to the notion of a *Hidden Markov Model*.

### 3.3 Hidden Markov Models

When listening to a voice, the sound one hears is the product of state changes in the vocal and nasal tracts, respectively. Variables include the size of the throat, position of the tongue and radiation effects at the lips. We thus have an observable speech signal and a hidden vocal system which are non-trivially related.

A hidden Markov model (HMM) extends the concept of a Markov chain by attaching an observation probability distribution to each state in the model. Within the framework of the theory, this effectively hides the states previously visible in Markov chains as each state has an ‘opinion’ about any observation value. We now have the ability to attach arbitrary semantics to model states which might not be readily accessible through observation in the process being modelled. This means that we can better model processes where we cannot directly observe the process states but instead have access to the outputs resulting from the internal state changes in the process. Model states do not necessarily have to correspond to some physical quantization of process states. Instead, they can be any abstraction with sensible semantics within the context of the process being modelled and the observations sampled from such a process. It is up to the modeller to decide on the semantics of a model. Sensible semantics will assist in initializing the model to startup values which will converge faster during discovery of the model parameters. The theory provides for a means to infer model parameters from a training set of observation sequences in a way which maximizes the likelihood of the sequences being generated by the model. Furthermore, we can calculate the likelihood that a sequence was generated by a model and derive the most probable state sequence corresponding to an observation sequence.

As stated, a hidden Markov model augments a Markov chain by coupling observation symbol distributions to the model states. As with Markov chains, we distinguish between

- *discrete models* where the process observations assumes one of a finite set of possible values. This calls for a probability distribution of the observations at each state.
- *continuous models* where the process observations are of continuous nature. This calls for a probability density function for the observations at each state.

A HMM is thus defined as

- $S$  : hidden states,
- $\Pi$  : starting state probabilities,
- $A$  : transition probabilities i.e.  $a_{ij} = P(q_t = S_j | q_{t-1} = S_i)$  and
- $B$  : observation symbol probability distributions i.e.  $P(o|S_i)$  or probability density functions  $p_s(o)$  where  $o$  is either a discrete or continuous variable.

The complete parameter set of the model is indicated by the compact notation

$$\lambda = (S, \Pi, A, B).$$

To extend the weather example we might imagine a scenario where the weather state is not observable any more. Instead we have access to readings from a barometer which measures atmospheric pressure. We can quantize such a reading into low, medium and high pressure to obtain discrete pressure values. We denote such a set of discrete observed values as  $V = \{v_1, v_2, \dots, v_M\}$  with  $M = |V|$  i.e.  $V_{weather} = \{low, medium, high\}$ .

Figure 3.2 shows a graphical representation of the Markov model extended to a HMM with the previously visible weather states now hidden *and* the barometer readings observable. The figure illustrates how probability distributions and density functions for discrete and continuous models, respectively, are coupled to states. The connection from a hidden state to an observable value represents the likelihood of generating the observable value, given that the model is in the hidden state. Thus, the likelihood of rain at a given time depends both

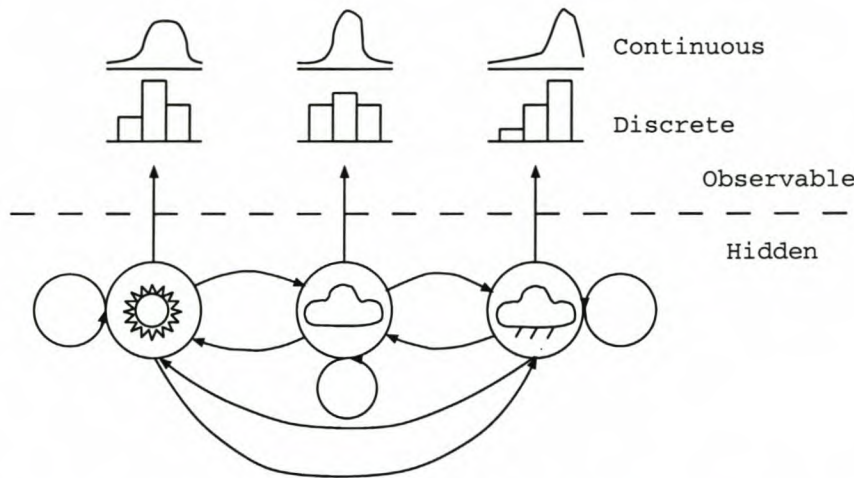


Figure 3.2: **Weather Hidden Markov Model** A hidden Markov model adds observation distributions to states of the Markov chain.

on the weather state at the previous time instance and the barometer reading for the current time instance. The observation probability distribution in state  $j$  can be expressed as

$$b_j(o_t) = P(o_t | q_t = S_j), \quad 1 \leq j \leq N.$$

In the discrete case, probabilities can be arranged in matrix form called a *confusion matrix* with a row assigned to each hidden state  $S_i$  and a column to each observable value  $o_t$ . In general, we will refer to the probability of generating symbol  $o_t$  when in state  $S_i$  as  $b_i(o_t)$ . The observable event for a particular hidden state is a stochastic variable and therefore the entries in any row of matrix  $B$  satisfy the condition

$$\sum_{j=1}^M b_{ij} = 1, \text{ for } 1 \leq i \leq N.$$

Many signals are continuous in nature and although a continuous signal can be quantized by one of the many vector quantization techniques available, information is likely to be lost in the process which could affect modelling performance to some extent. It is hence advantageous to have HMMs with continuous observation density functions such as Gaussian mixtures. In this case, we then have the condition

$$\int_{-\infty}^{\infty} b_j(x) dx = 1, \text{ for } 1 \leq j \leq N.$$

When working with continuous HMMs, an assumption about the form of the density function has to be made. Some studies [36] prefer to discretize continuous values to avoid making this

assumption. Another approach is to employ neural networks to learn the form of the density functions [28].

This weather model can be used to answer questions regarding various weather related issues. For instance, we want to use the model in such a way as to determine what the most probable season is in which a certain string of barometer measurements were made. This brings us to 3 problems associated with HMMs as generally itemized in the literature:

1. Given an observation sequence  $O = o_1 o_2 \dots o_T$  and a HMM  $\lambda$ , how do we **efficiently** compute  $P(O|\lambda)$  i.e. the probability of the observation sequence being generated by the model?
2. Given an observation sequence  $O = o_1 o_2 \dots o_T$  and a model  $\lambda$ , what state sequence  $Q = q_1 q_2 \dots q_T$  best explains the observations? Several optimality criteria exist and the appropriate one to use depends on the problem at hand.
3. Finding a suitable set of parameters  $\lambda$  to model a process. The question now arises what the word *suitable* suggests. We would like to maximize the probability  $P(O|\lambda)$  where  $O$  is an observation sequence sampled from the process being modelled and forms part of a set of training sequences to deduce  $\lambda$  from.

### 3.3.1 Problem 1: Observation Sequence Likelihood

We want to compute the likelihood of the observation sequence  $O$  of length  $T$  being generated by the model  $\lambda$  i.e.  $P(O|\lambda)$ .

Figure 3.3 illustrates the observation sequence *low low medium high* and the possible hidden states at each time instance as a trellis. The most straightforward way of calculating  $P(O|\lambda)$  is by enumerating each of the  $N^T$  possible state sequences of length  $T$  and summing, which is of exponential complexity in the order of  $O(TN^T)$ . This led to the development of what is known as the *forward procedure*. The procedure is made possible by the time invariance of the probabilities in HMMs.



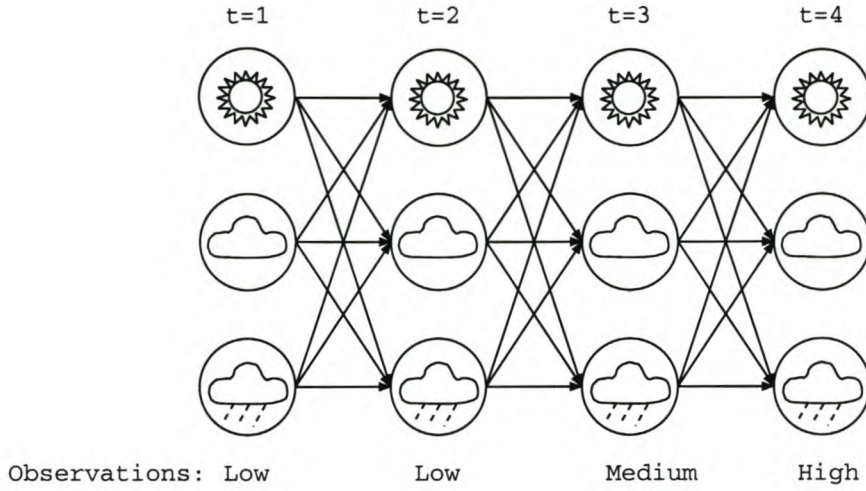


Figure 3.3: **Forward Procedure** Path likelihoods of the partial observation sequence up to a certain time are calculated.

We define a variable (called the *forward* variable) as

$$\alpha_t(i) = P(o_1 o_2 \dots o_t, q_t = S_i | \lambda),$$

which is the probability of the partial observation sequence  $o_1 o_2 \dots o_t$  and the model  $\lambda$  being in state  $S_i$  at time  $t$ . An inductive calculation of  $\alpha_t(i)$  is

1. Basis for induction

$$\alpha_1(i) = \pi_i b_i(o_1), \text{ for } 1 \leq i \leq N.$$

The forward probabilities are initialized to be the joint probability of starting out in state  $S_i$  and the first observation symbol being  $o_1$ .

2. Inductive step

$$\alpha_{t+1}(j) = \left[ \sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), \text{ for } 1 \leq t \leq T - 1 \text{ and } 1 \leq j \leq N.$$

This step expresses the fact that any hidden state in the vertical columns of the trellis of Figure 3.3 can only be reached via the  $N$  hidden states in the previous column, i.e. previous time instance. From this, we deduce that to any of the  $N$  hidden states at time  $t$ ,  $1 \leq t \leq T$ , there exist  $N^{t-1}$  distinct state sequences or paths leading to this state from the states at time  $t = 1$ . This is illustrated in Figure 3.4. Due to the time invariance of  $A$  and  $B$ , we can use induction to calculate a forward variable for a specific

state at a specific time rather than trace all the possible sequences the entire way back to time  $t = 1$ . We therefore calculate  $\alpha_{t+1}(j)$  as the sum of the forward variables for the partial observation sequences up to time  $t$  over all the hidden states, each time multiplying by the transition probability for being in the hidden state and making a transition to state  $S_j$  (which is the terminal state for the  $\alpha$  value being calculated). This sum is then multiplied by the observation probability for symbol  $o_{t+1}$  observed when in state  $S_j$ .

### 3. Termination

$$P(O|\lambda) = \sum_{i=1}^N \alpha_T(i).$$

The desired probability is given by the sum of the terminal forward variables  $\alpha_T(i)$ .

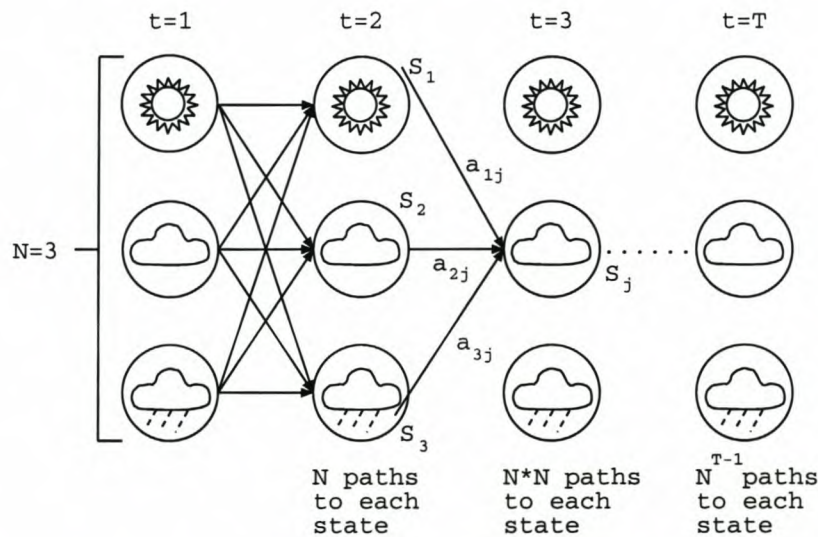


Figure 3.4: **State Sequences** The number of possible paths increases as a power of the sequence length.

The time complexity of this procedure is in the order of  $O(N^2T)$  which is a huge improvement over  $O(TN^T)$ .

### 3.3.2 Problem 2: Most Probable State Sequence (Viterbi Algorithm)

We want to find a hidden state sequence which best explains the observation sequence. There is no exact solution to this problem (as there is to problem 1) due to the uncertainty of which

optimality criterion to use. The most popular approach is called the *Viterbi* algorithm which stems from dynamic programming methods. This algorithm attempts to find a single best hidden state sequence through the graph of Figure 3.3 every time inspecting  $b_i(o_t)$  to make its decisions. To classify a sequence as being *best*, it needs to be enumerated and score a higher value than all other sequences in the graph. A hidden state sequence  $Q$  is enumerated for observation sequence  $O$  through

$$\pi_{q_1} b_{q_1}(o_1) \prod_{t=2}^T a_{q_{t-1}q_t} b_{q_t}(o_t)$$

This action is equivalent to maximizing  $P(Q|O, \lambda)$ . For each intermediate and terminating state in the trellis of Figure 3.3 there is a most probable path to that state. So, for example, each of the three states at  $t = 4$  will have a most probable path to it, perhaps as in Figure 3.5.

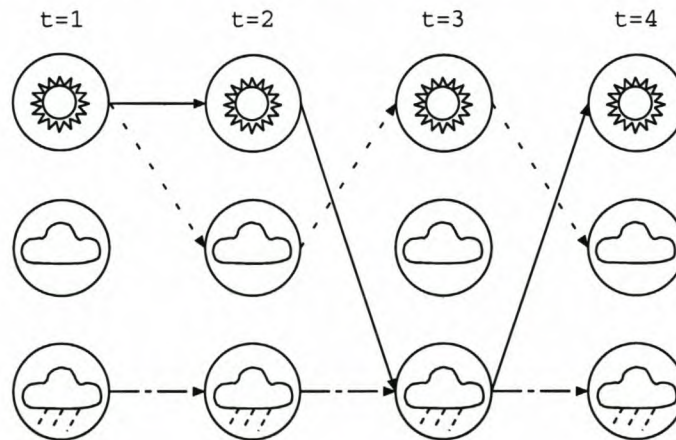


Figure 3.5: **Possible state sequences** Each state may have a most probable path ending in that state. We take the path with the highest likelihood.

The Viterbi algorithm, rather than enumerating all possible sequences in a brute force fashion and selecting the maximum scoring one, goes about recursively to find the sequence as follows.

We define the quantity

$$\delta_t(i) = \max_{q_1 \dots q_{t-1}} P[q_1 \dots q_t = S_i, o_1 \dots o_t | \lambda]$$

which is the best scoring sequence of length  $t$  ending in state  $S_i$ . By induction we have

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(o_{t+1})$$

This quantity will facilitate finding the highest score; however, the state chosen to maximize the quantity at each time instance, needs to be remembered if the actual sequence is to be reconstructed after the search terminates. For this purpose, an array  $\psi_t(j)$  is used. The complete recursive algorithm then is

1. Basis for induction

$$\delta_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N \quad (3.1)$$

$$\psi_1(i) = 0 \quad (3.2)$$

2. Inductive step

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(O_t), \quad 2 \leq t \leq T, \quad 1 \leq j \leq N \quad (3.3)$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \quad 2 \leq t \leq T, \quad 1 \leq j \leq N \quad (3.4)$$

3. Termination

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \quad (3.5)$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \quad (3.6)$$

4. Sequence reconstruction

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \text{ for } t = T - 1, T - 2, \dots, 1 \quad (3.7)$$

The Viterbi algorithm provides a computationally efficient way of analysing observations of HMMs to recapture the most likely underlying state sequence. It exploits recursion to reduce computational load, and uses the context of the entire sequence to make judgements, thereby allowing a good analysis.

### 3.3.3 Problem 3: Baum-Welch Parameter Re-estimation

We want to adjust the model parameters  $\lambda$  to maximize the probability  $P(O|\lambda)$  for a given  $O$ . This is a learning problem and for any finite observation sequence as training data, we can estimate  $\lambda$  so that, at best,  $P(O|\lambda)$  will be locally maximized. This section will present a re-estimation procedure for iteratively updating and improving  $\lambda$ , called the *Forward-Backward* or *Baum-Welch* algorithm. This re-estimation procedure is based on the principle of maximum likelihood estimation (MLE). With MLE, the parameter(s) describing a likelihood function (i.e. the parameters of the HMM) are discovered by holding fixed the underlying random variable and varying the function parameters in such a way as to maximize the function. The Baum-Welch re-estimation formula described here converges to a parameter set  $\lambda$  for an HMM which maximizes the function [11]. Any good mathematical statistics textbook can be consulted for a description of MLE [25].

We define a variable (called the *backward* variable) as

$$\beta_t(i) = P(o_{t+1} \dots o_T | q_t = S_i, \lambda)$$

which is the probability of the partial observation sequence from  $t + 1$  to the end, given the model  $\lambda$  is in state  $S_i$  at time  $t$ . As with the forward variable, we can solve for  $\beta_t(i)$  inductively as follow

1. Basis for induction

$$\beta_T(i) = 1, \text{ for } 1 \leq i \leq N$$

2. Inductive step

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \text{ for } t = T - 1 \dots, 1, \text{ and } 1 \leq i \leq N.$$

Furthermore, we define

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda)$$

which is the probability of being in state  $S_i$  at time  $t$  and state  $S_j$  at time  $t + 1$  given the model  $\lambda$  and observation sequence  $O$ . We can write  $\xi_t(i, j)$  in terms of the forward variable

introduced during the solution to problem 1 and the backward variable as follows

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}$$

The denominator normalizes the term into a probability measure. Figure 3.6 graphically illustrates the calculation with  $a_{ij} b_j(o_{t+1})$  providing the link between the partial sequence probability up to time  $t$  ending in state  $S_i$  and the partial sequence probability from time  $t + 1$  onwards starting in state  $S_j$ .

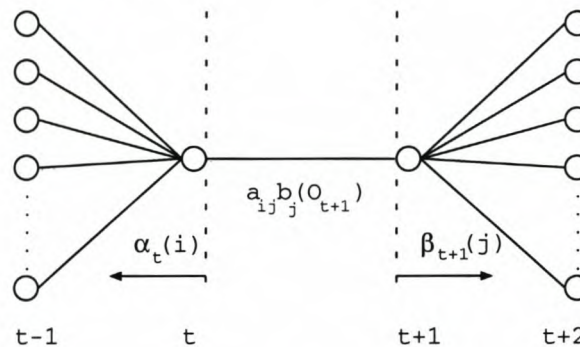


Figure 3.6: **Forward-Backward procedure** The forward  $\alpha$  and backward  $\beta$  variables are used to calculate the likelihood of a state sequence being in a certain state at a certain time.

Having  $\xi_t(i, j)$ , we define

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j)$$

which is the probability of being in state  $S_i$  at time  $t$ , given  $O$  and  $\lambda$ . Summing  $\gamma_t(i)$  over time gives the expected number of times that state  $S_i$  will be visited or, when time  $t = T$  is excluded, the expected number of transitions from state  $S_i$

$$\Gamma(i) = \sum_{t=1}^{T-1} \gamma_t(i)$$

Likewise, summing  $\xi_t(i, j)$  over time gives the expected number of transitions from state  $S_i$  to  $S_j$

$$\Xi(i, j) = \sum_{t=1}^{T-1} \xi_t(i, j)$$

Using the defined quantities, the Baum-Welch algorithm re-estimates the HMM parameters as follows:

$$\bar{\pi} = \gamma_1(i)$$

$$\overline{a_{i,j}} = \frac{\Xi(i,j)}{\Gamma(i)}$$

$$\overline{b_i}(k) = \frac{\sum_{t=1}^T \text{and } O_t=v_k \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)}$$

If  $\lambda$  did not already define a critical point of the likelihood function, in which case the re-estimation will have no effect, the new parameter set  $\bar{\lambda}$  obtained from the procedure, describes a model more likely to have produced the observation sequence  $O$ . The procedure preserves stochastic constraints for the HMM parameters namely

$$\sum_{i=1}^N \overline{\pi_i} = 1,$$

$$\sum_{j=1}^N \overline{a_{ij}} = 1, \text{ for } 1 \leq i \leq N \text{ and}$$

$$\sum_{k=1}^M \overline{b_j}(k) = 1, \text{ for } 1 \leq j \leq N.$$

Unfortunately, some difficulties arise when implementing HMMs in finite computing systems. The calculations involved when working with HMMs, often multiplies probabilities which are by definition less than 1. When the length of a sequence is large enough, the computed values generally decrease beyond the precision range of most computing machines. A procedure does exist which scales calculated values to fall within a computable range and results in probabilities in the log domain. This procedure can be found in [56] and will not be presented here. Instead, we adopt another training procedure based on the Viterbi algorithm which provides an easier way to overcome this problem. The algorithm has the added advantage that it enables faster training than the Baum-Welch re-estimation procedure.

### 3.4 Viterbi Training Procedure

The Viterbi algorithm described previously in Section 3.3.2 finds the most probable state transition sequence a process undergoes whilst generating a particular observation sequence.

The forward and Viterbi algorithms together with a hidden Markov model configuration both define probability density functions over a space of observation sequences. The algorithm can thus calculate the likelihood that an observation sequence was generated by a model. The likelihood calculated by the Viterbi algorithm differs from that calculated by the forward algorithm as explained in Section 3.3.1. This suggests that the overall shape of the probability density function defined by the Viterbi procedure differs slightly from that defined by the forward procedure. Thus, the Viterbi training algorithm maximizes the likelihood of the training set for a different density function than does the Baum-Welch procedure. Experience has shown, however, that the results obtained by using this approach do not significantly differ from those obtained by using the Baum-Welch algorithm; yet, they require far fewer computations.

Before explaining the training algorithm, we show how the computations of the Viterbi algorithm are modified to overcome the problem of values exceeding the precision range of computers [56, 36]. The idea is to perform calculations in the log domain. The Viterbi algorithm is modified by changing Equation 3.1 to

$$\delta_1(i) = \log(\pi_i) + \log(b_i(O_1)), \text{ for } 1 \leq i \leq N.$$

Equation 3.3 becomes

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) + \log(a_{ij})] + \log(b_j(O_t)), \text{ for } 2 \leq t \leq T \text{ and } 1 \leq j \leq N.$$

This results in the calculation of a log-likelihood with Equation 3.5 becoming

$$\log(P^*) = \max_{1 \leq i \leq N} [\delta_T(i)]$$

Once a model configuration has been decided on, the model parameters need to be discovered from a training set of sequences. Prior knowledge about the problem domain and the semantics of the model states assist in initialization of the model parameters before training commences. This often proves to be crucial in achieving a good representation of the modelled process. It is fairly common for transition variables to be initialized to random values maintaining stochastic constraints. However, applying this approach to set initial values of the probability distributions/density functions is most likely to result in far less than optimal



parameter discovery. In Chapter 4, various initialization schemes are employed prior to training models. Initialization can be seen as biasing the training procedure and the challenge thus resides in finding a good model bias for the problem at hand. A description of the Viterbi training algorithm is now presented.

The Viterbi algorithm is conducted in a batch fashion meaning that all the training sequences are used during a single re-estimation iteration. From the training sequences  $O_K$  we re-estimate  $\lambda$  by

- the starting probability for state  $S_i$  as

$$\bar{\pi}_i = \frac{\sum q_1^k = S_i}{K}, \text{ for } 1 \leq i \leq N$$

i.e. the number of times a computed state sequence starts in state  $S_i$  as a fraction of the number of training patterns,

- the transition probability for state  $S_i$  to  $S_j$  as

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} q_t^k = S_i \text{ and } q_{t+1}^k = S_j}{\sum_{t=1}^{T-1} q_t^k = S_i}, \text{ for } 1 \leq i, j \leq N,$$

i.e. the number of times a transition is made from state  $S_i$  to  $S_j$  as a fraction of the number of times state  $S_i$  was visited and

- the observation symbol probability (in the discrete case) for state  $S_i$  and observation symbol  $v_j$  as

$$\bar{b}_{ij} = \frac{\sum_{t \forall q_t^k = S_i} o_t = v_j}{\sum_{t=1}^{T-1} q_t^k = S_i}, \text{ for } 1 \leq i \leq N, 1 \leq j \leq M,$$

i.e. the total number of times the model was in state  $S_i$  and generated observation symbol  $v_j$  as a ratio of the number of times  $S_i$  was visited. In the continuous case the observation value  $v_j$  contributes to an average observation value for state  $S_i$  and a second traversal of the sequences is needed to determine the standard deviation from this average. These values are then used to define a probability density function.

During training, we have the option of imposing a restriction on the allowed terminal state used by the alignments. This means we can say that an alignment may not terminate further to the left from the rightmost state (in the case of left-right models) than a preset distance.

Our experience has shown that this approach results in models with a better ability to distinguish between true and false exemplars. The Viterbi training algorithm is also described briefly in [36].

As can be seen, the Viterbi training algorithm is straight forward and contains less calculations than the Baum-Welch algorithm. It should be noted that this algorithm is based on the assumption that a most probable state sequence can be matched to an observation sequence. This means that, prior to training, the model needs to be initialized in such a way that from the outset, training sequences are close enough to model state observation distributions to prevent underflow due to uncomputable likelihood values. This stresses the importance of a good biasing initialization prior to the re-estimation training procedure.

### 3.5 State Transition Configurations

As explained in Section 3.2, a model is considered *ergodic* when any state in the model can be reached from any other state in a single transition. This however need not always be the case. In a *left-right* (Bakis) HMM [56, 36, 71], states are numbered in ascending order; a system either remains in the same state, i.e. a self-transition, or it transitions to a state with a higher index. In such a model, the initial state probabilities,  $\Pi$ , have the property that only the left-most state has a non-zero starting probability i.e.  $\pi_0 = 1$  and  $\pi_{1..S} = 0$ . This type of model has been found to account for the observed properties of certain types of signals better than does the ergodic model. In a variant of this model, parallel paths through the model are also allowed. Figure 3.7 shows a graphical depiction of these types of models. Conceptually, any configuration is possible and will have no influence on the re-estimation procedure. State transitions set to zero when re-estimation commences, will remain zero. The converse is not true however, meaning non-zero transitions could very well become zero as the re-estimation procedure iterates.

Figure 3.7 helps us to understand the conceptual difference between left-right and ergodic models. With Left-right models, the observation density functions for states at certain offsets from the left of the state graph, correspond closely to actual sequence values at similar offsets from the beginning of the sequence. A left-right model thus acts as a sequence memory

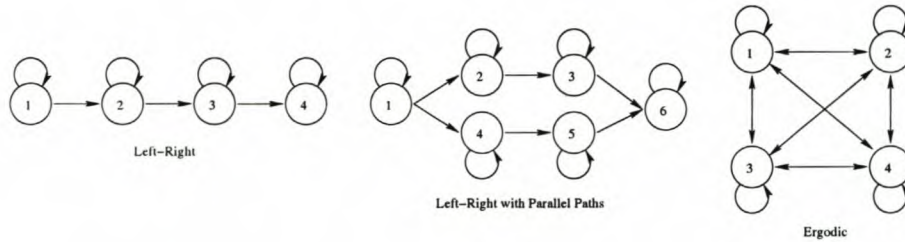


Figure 3.7: **Types of hidden Markov models:** Different types of hidden Markov models are defined by the state transition configurations.

allowing only for marginal deviations from a representative sequence both in sequence values (vertical deviations) and timing information (horizontal deviations). We will thus use left-right models when we want to model a signal source which produces fairly similar sequences i.e. stationary processes. Ergodic models, on the other hand, allow for self-similarities within the signal as any state in the model can be reached at any stage. This allows for better generalization; however, the set of sequences which will match the model is not as intuitively predictable as is the case with left-right models. Ergodic models can thus match unseen sequences disparate from those in the training set.

### 3.6 Duration Modelling

As described in Section 3.2, the state duration probability density function is

$$P(O|M, q_1 = S_i) = 1 \cdot (a_{ii})^{d-1} (1 - a_{ii}) = p_i(d)$$

based on the value of the self-transition probability  $a_{ii}$  of a state. This density function decays exponentially and is not suitable for many physical signals. An explicit way of modelling state duration is presented in [56]. This, however, results in a quadratic increase in computational cost. An alternative heuristic which alters the computed log-likelihood in a postprocessing phase according to duration probability histograms derived from a segmental K-means procedure can be used instead.

Duration modelling is achieved in [36] by introducing a number of unit-duration sub-states replacing each original model state. The transitions from one state to the sub-states of the

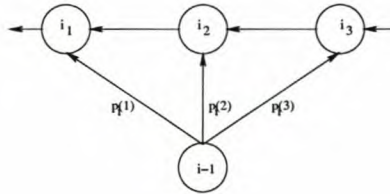


Figure 3.8: **Unit Duration Sub-states:** Duration modelling can be approximated by decomposing a HMM state into unit- duration sub-states.

next state approximates the wanted transition probability density function. This approach has the disadvantage that it significantly increases the number of model parameters. Figure 3.8 shows an example of such a substitution.

In Chapter 4, we explore a simple way of limiting the maximum number of self-transitions in a state by restricting the Viterbi algorithm to allow only a specified number of self-transitions in each separate state. This is realized by maintaining a duration count for each state and updating this parameter along with the other model parameters during re-estimation. With each re-estimation iteration, the duration count for a given state is decremented by one if none of the training sequences result in a Viterbi alignment which remains in that state for exactly the allowed count. If, however, there is a state sequence which remains in the state for the maximum allowed count, the allowed duration count is incremented by one. This does not achieve the same effect as the duration modelling described in the previous paragraphs. It does however enable a model to disallow sequences which scores high due to the entire sequence consisting of values close to the mean of a single state's observation density function.

In our application of HMMs in Chapter 4, the possibility to restrict large deviations from the training sequences used to build the models, is an important factor. The ability to limit the number of times a state may be repeated in a highest probability state sequence, prevents an alignment with sequences which match the observed values in certain states for excessive periods of time. Together with the restriction on the allowed terminal state, this form of duration restriction forces a sequence to more or less conform to the profile of the entire

length of training sequences. It will prohibit sequences which deviate from such a model to score high likelihoods when the alignment is calculated.

### 3.7 Other Possible Extensions

A number of other extensions to HMMs have also been proposed particularly as far as recognition of complex gestures are concerned [14, 66]. Many hybrid combinations of HMMs and Artificial Neural Networks (ANNs) have also been proposed in the field of automatic speech recognition [13, 12, 60, 73, 35, 63, 17]. It is shown how ANNs can be used to learn the observation symbol distributions in states, relieving us from having to make assumptions about the shape of the distributions. Furthermore, ANNs are used to perform input transformations on the data before presenting it to a HMM. Gradient descent learning rules adapted from ANNs have also been applied to HMMs in an attempt to create a unifying learning method to better integrate HMMs and ANNs.

### 3.8 Summary

Hidden Markov models are a statistical modelling tool for time series modelling. They have been applied to various fields of research with a great deal of success. This chapter provided the necessary background on hidden Markov models to understand the next chapter which applies HMMs to automatic signature verification. HMMs are well documented and although we chose to implement our own HMM toolkit, several publicly available HMM toolkits exist for download from the Internet.

## Chapter 4

# A Signature Verification System

### 4.1 Introduction

Hidden Markov models (HMMs) have been applied to signature verification in the past with varying degrees of success [71, 23, 36, 48]. In this chapter, we describe a dynamic signature verification system developed at the Computer Science Department of the University of Stellenbosch. This system makes use primarily of hidden Markov models (HMMs) to measure the process of handwritten signature rendition. A number of different configurations are tested in search of a model which enables the automatic verification of signatures with accuracy as high as possible.

A HMM is a generic tool for sequence modelling. Modelling the output from processes with HMMs calls for two important considerations to be contemplated: (1) what the meaning coupled to model hidden states will be, (2) what information an observation sequence characterizing a process should contain. The models described in Section 4.5 explore a number of possible model configurations. The state semantics often dictate the number of states in a model. This becomes an important factor when considering a system for commercial use as the number of states are proportional to the amount of computing resources needed to perform the task. The observable aspects available to describe a signature depend largely on the hardware used to capture signatures. From these sampled signals, other signals can be deduced which may be more directly descriptive of the signing process [72]. The components

in an observation sequence are often closely tied to the semantics of the model states.

Preprocessing serves an important purpose in attaining acceptable performance levels in a verification system [54]. Varying hardware specifications and constraints on the data imposed by the modelling technique often require a comprehensive preprocessing phase. Some studies perform little or no preprocessing [50]. Our system subjects signatures to a number of priming actions prior to the actual modelling phase. Each preprocessing action has a distinct purpose which will be described in Section 4.3. For other preprocessing methods encountered in the literature, please consult Section 2.4.

Finally, the credibility of experimental results for verification systems depends largely on the quality of the signature database used to obtain the results. A signature database has to be as representative of the intended target audience as possible. The quality of forgeries is another very important aspect when measuring system performance. Section 4.7 describes the database used in our experiments. The results for these experiments are presented in Chapter 5.

## 4.2 Signature Acquisition

We used a WACOM Intuos [4] graphical tablet to capture signatures. Figure 4.1 shows an image of the tablet. Data is sampled at a rate of 200 Hz. Each sample consists of

- the current  $x$  and  $y$  position of the pen tip on the surface of the tablet
- The pressure the pen tip exerts on the tablet surface quantized to 64 levels
- The angle the pen makes with the  $x$  and  $y$  axis respectively

These raw signals are plotted in Figure 4.2 together with the signature they were sampled from. The tablet has a resolution of 1000 lpi. For further details on the data acquisition phase, please see Section 4.7.

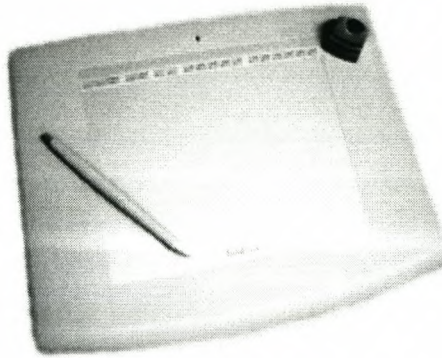


Figure 4.1: **WACOM Graphical Tablet:** This device is used to measure various aspects of the signing process.

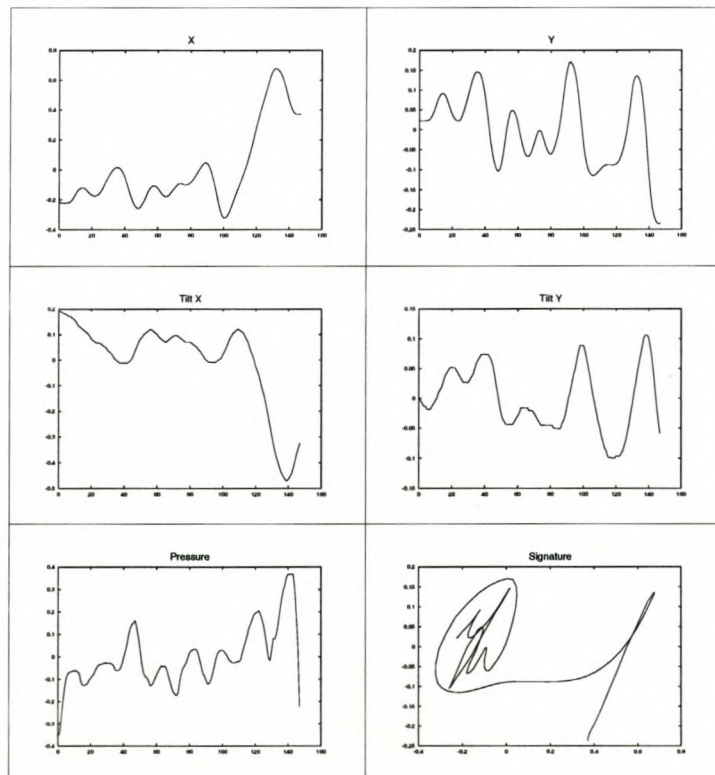


Figure 4.2: **Sampled Signature Components:** Positional, pressure and tilt information are reported by the tablet.



### 4.3 Signature Preprocessing

The raw sequences of signature components are not in a form which is suitable for modelling. The HMMs will require signatures to be in a canonical form prior to training and verification. Samples of a signature can be transformed differently by rotation, translation and scaling when initially sampled from a user. The preprocessing actions seek to convert a raw signature into canonical form with respect to orientation. Once in this form, we can include the signature in either the training database or subject it to verification.

The ideal scenario is for a system not to restrict a writer to sign on a certain baseline. Inspection of a couple of signatures revealed that even if a baseline is provided, signatures are not guaranteed to follow them. Some writers start signing on the baseline but progress in a direction pointed to the top right of the signing space. It is arguable whether this imaginary baseline can be assumed to be constant. Furthermore, signatures seldom start exactly at the beginning of the baseline. Generally speaking, they might start anywhere in the first quarter of the line. The size of signatures also vary from one exemplar to another. These factors opt for some procedure to convert a signature into a uniform reference frame. Section 2.4 describes some approaches to solve this problem as found in the literature. The algorithm described in this section is essentially an elaboration on our method reported in [65].

#### 4.3.1 Rotational Invariance

Rotational invariance is achieved by calculating an angle  $\theta$  of corrective rotation about the centroid of the  $(x, y)$  samples. Rotating the signature by  $\theta$  normalizes it to a line running through the centroid. We calculate  $\theta$  by maximizing the deviation of the data in one direction, e.g. the  $x$  direction <sup>1</sup> The normalized signature is obtained as follows:

The mean  $\mu_x$  of the  $x$  sequence is calculated by

$$\mu_x = \frac{\sum_t^T x_t}{T}$$

---

<sup>1</sup>The normalized signature is not needed for display purposes. Thus, we may alter it in any consistent way we wish.

and the standard deviation of the  $x$  sequence from  $\mu_x$  by

$$\sigma_x = \sqrt{\frac{\sum_t^T (x_t - \mu_x)^2}{T}}$$

In order to maximize the deviation, we need only to maximize

$$\sum_t^T (x_t^* - \mu_x)^2$$

where  $x_t^*$  indicates a rotated  $x$  value. We need to choose a point which is rotational invariant within the framework of the normalization scheme [67]. It is easy to show that the centroid is such a point within this scheme. A rotation about the centroid  $(\mu_x, \mu_y)$  can be expressed as

$$x_t^* = (x_t - \mu_x) \cos \theta + (y_t - \mu_y) \sin \theta + \mu_x$$

By substituting  $x_t^*$  into the expression to be maximized, we obtain  $f(\theta)$  as

$$\begin{aligned} f(\theta) &= \sum_t^T [(x_t - \mu_x) \cos \theta + (y_t - \mu_y) \sin \theta + \mu_x - \mu_x]^2 \\ &= \sum_t^T a_t^2 \cos^2 \theta + 2a_t b_t \cos \theta \sin \theta + b_t^2 \sin^2 \theta \\ &= \cos^2 \theta \sum_t^T a_t^2 + 2 \cos \theta \sin \theta \sum_t^T a_t b_t \\ &\quad + \sin^2 \theta \sum_t^T b_t^2 \\ &= P \cos^2 \theta + 2Q \cos \theta \sin \theta + R \sin^2 \theta \end{aligned}$$

where

- $a_t = x_t - \mu_x$ ,
- $b_t = y_t - \mu_y$ ,
- $P = \sum_t^T a_t^2$ ,
- $Q = \sum_t^T a_t b_t$  and
- $R = \sum_t^T b_t^2$ .

Taking the derivative yields

$$f'(\theta) = 2Q \cos^2 \theta - 2P \cos \theta \sin \theta \\ + 2R \cos \theta \sin \theta - 2Q \sin^2 \theta$$

with roots

$$\pm \arccos\left(\pm \frac{1}{\sqrt{2}} \sqrt{\frac{1 \pm (P - R)}{\sqrt{P^2 + 4Q^2 - 2PR + R^2}}}\right)$$

We adopt the value for  $\theta$  closest to zero which will result in a maximum of  $f(\theta)$ .

We also need to ensure that the time series evolves in a consistent direction by imposing a possible 180° rotation. This is done by fitting a least squares line to the rotated  $x$  data. If the slope is less than 0, we infer that the signature strokes are increasing from right to left and not left to right as is the case with normal signers. We then apply an additional 180° rotation to conform to the norm. This means that the tablet can be upside-down when signing without affecting the normal operation of the system.

Unfortunately, there does exist a scenario where this scheme fails. If a writer's signature is a borderline case where the maximum deviation varies from one axis to the other with different samples, this scheme will result in an inconsistent perpendicular normalization. Fortunately, these signatures are rare as signers usually sign in a predominantly left-to-right fashion. One partial solution to this problem is achieved by providing a baseline. Now, when a  $\theta$  value larger than some acceptable deviation from the baseline, e.g. 45°, is attained, we conclude that the algorithm is confused by a borderline case as explained earlier. In such a case we have to trust the writer blindly and perform no rotation. Another solution is to calculate the ratio of variances in the  $x$  and  $y$  directions. If this ratio is within a threshold distance from 1.0, we conclude that the signature is not suitable for normalization by this technique.<sup>2</sup>

The pressure signal is invariant to the rotation of a signature baseline. However, it seems that a common oversight is the pen tilt signal which needs to be transformed along with the positional information. The tablet reports the tilt as the angle the pen makes with the  $x$  and  $y$ -axes on the tablet surface. To understand why a rotation might affect these tilt angles, imagine the pen coinciding with the surface of a cone where the cone tip is situated at the

<sup>2</sup>For a literature survey explaining common problematic signature types see [30, 16].

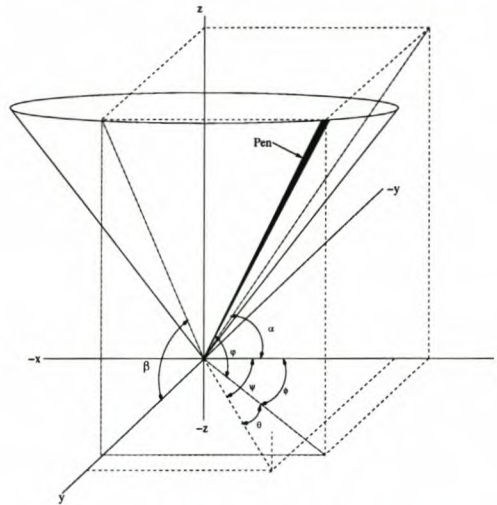


Figure 4.3: **Pen Tilt Rotation:** The reported pen tilt is not invariant to baseline rotation.

pen tip. A rotation of the coordinate system (i.e. the tablet surface) will result in a swept cone. To calculate the new tilt angles, we create a top-view of the pen using the reported tilt angles. The pen is then rotated by the angle  $\theta$  calculated in the previous paragraph. Images of this rotated pen are then projected back onto the the  $xz$ - and  $yz$ -planes, respectively, to calculate the new angles. Figure 4.3 graphically depicts this process. In the figure,  $\alpha$  is the angle formed with the  $x$ -axis and  $\beta$  the angle formed with  $y$ -axis. To transform  $\alpha$  and  $\beta$ , we project the image of the pen as seen from above, onto the  $xy$ -plane resulting in the pen angle  $\phi$  in the figure. This projected pen vector is then rotated by the normalization angle  $\theta$  mentioned previously, resulting in a pen vector with angle  $\psi$ . From this vector, we project the image of the pen onto the  $xz$ - and  $yz$ -planes, respectively,, respectively, and calculate the new values for  $\alpha$  and  $\beta$ .

### 4.3.2 Translation Invariance

To compensate for the fact that a signer need not always start on the exact same place on a baseline, we apply a translation to the signature. After this operation, the leftmost part of the signature will coincide with the vertical axis of the two-dimensional Cartesian plane and the bottommost part with the horizontal axis. The translation vector is simply taken to be the smallest coordinate value in the signature on both axis. This vector is then subtracted from each  $(x, y)$  sample.

### 4.3.3 Scaling Invariance

To achieve scaling invariance, we need to find a scaling factor which will transform the size of the signature to be contained in a 1-by-1 box yet maintain its aspect ratio. The dimensions of the box are arbitrary as we merely need a unifying signature size. The scaling factor is taken as the minimum of  $\frac{1}{X_{dim}}$  and  $\frac{1}{Y_{dim}}$  and both the  $x$  and  $y$  components are multiplied by this factor. This operation maintains the original aspect ratio.

### 4.3.4 Acquisition Device Invariance

Even though there was no need for this step to be performed in our experimental system, it will be important for a production system to be sure that differences in hardware do not hamper the performance of the system. The software drivers on different platforms may also translate the sampled values to different intervals than what is reported by the hardware.

As tablet brands may quantize the pressure signal differently, we need to convert the pressure values to a uniform interval. The default tablet we use in our system reports the pressure in a range of  $[0; 63]$ . We scale these values to reside within the interval  $[0; 1]$ .

The angle of tilt is reported to be in the range  $[-1; 1]$  by the software drivers used in the graphical user interface described in Section 4.7. On the other hand, the benchmark database also described in that section records the tilt in degrees. As we interpret the tilt value as an explicit angle during pre-processing, we opt to convert the sampled value to  $[-90^\circ; 90^\circ]$ .

Some systems might require for uniform sampling rate as signature duration is seen as an important feature. If a signature is obtained from a tablet at a different rate than what was used to derive the model parameters, it might need to be resampled to conform to a unifying standard. To understand why this is important, we visualize a scenario where the system is used in an open commercial environment. Different transaction end-points might deploy different tablet brands. For the system to function correctly across different platforms, it might be necessary to agree on a uniform sampling frequency. Resampling to such a frequency can be performed by fitting an interpolating spline to the data. We make use of relative durations wherever timing information is required. Our final signature likelihood

values are also normalized with respect to duration as is described in Section 4.6.2. We therefore do not need to perform this action.

#### 4.4 Arc-length Parameterization

The arc-length parameterization is the preferred means of reference to signatures in [51, 50]. As we sample signatures from a tablet at a fixed sampling rate, we have a time parameterization of the signature signals. Apart from this parameterization, we would also like to conduct experiments on the arc-length parametrized versions of the signatures.

The arc-length parameterization of a curve can be constructed from another differentiable parameterization by the following process:

1. the cumulative arc-length of a parameterized curve  $\bar{r}(t) = (x(t), y(t))$  measured from  $t = a$  is given by

$$l(t) = \int_a^t \|\bar{r}'(\tau)\| d\tau = \int_a^t \sqrt{x'^2(\tau) + y'^2(\tau)} d\tau$$

2. the inverse of the arc-length function is used to create an arc-length parameterization of a curve by composition

$$s(u) = (x \circ l^{-1}(u), y \circ l^{-1}(u))$$

A necessary and sufficient condition for a curve to be in arc-length parameterization form is

$$l'(t) = 1 \quad \forall t$$

By taking the derivative of  $l(t)$  and the previous condition we arrive at another condition for the arc-length parameterization

$$x'^2(t) + y'^2(t) = 1$$

from this we can see that

$$|\bar{r}'(t)| = 1$$

which means that the arc-length parameterization describes the traversal at unit-speed of the curve.

Unfortunately, there are some difficulties in obtaining an arc-length parameterization for the signatures. This is due to the fact that  $l^{-1}$  for polynomial curves of degree  $n \geq 3$  such as the cubic splines we use for interpolation, cannot in general, be expressed as any elementary function. One solution to the problem is to use a different class of interpolating functions which do lend themselves to arc-length parameterization. In [29] amongst others, we find definitions of such curves. We opt however for a numerical approximation proposed in [21]. This approximation uses an adaptive sampling of parametric curves with respect to local curvature. In regions of high curvature, the sampling density is increased and *vice versa*. These samples can now be used to approximate the curve length by accumulating the direct distances between samples. The adaptive sampling rate ensures that nowhere on the curve will a straight line between two points deviate unacceptably far, i.e. further than a certain threshold value from the actual curve. The general method is based on the following strategy:

1. choose a flatness criterion to be used for refinement
2. evaluate the criterion on an interval on the curve
3. if the interval is flat enough i.e. below/above some threshold, then the interval can be represented by the two extremes
4. otherwise, recursively subdivide the interval until no sub-interval violates the flatness criteria and represent the interval by this set of samples

Various flatness criteria can be used during probing where an intermediate point  $v_m$  is chosen between the two extrema  $v_a$  and  $v_b$ .

- area of a triangle formed by  $v_a v_m v_b$  is small

- the angle formed by  $v_a v_m v_b$  is close to  $180^\circ$
- the length  $|v_a - v_m| + |v_m - v_b| \approx |v_a - v_b|$

We use the angle formed by  $v_a v_m v_b$  as flatness criteria. The pseudo code for a recursive implementation of the length function can now be given as:

```

length(a, b, va, vb)
  m ← random point in [a,b]
  vm ←  $\gamma(m)$ 
  if flat(va, vm, vb)
    return |vb-va|
  else
    return length(a, m, va, vm)+length(m, b, vm, vb)

```

where  $\gamma(t)$  is the function to sample from, which in our case is a cubic spline.

This length function is used to map the time indices of the signature as sampled, to arc-length. The length function together with this mapping are then used in a binary search procedure to find the exact time parameter (or rather within a certain toleration) corresponding to a specific arc-length parameter. Figure 4.4 shows the discrete sample points of a time and arc-length parameterization of a signature. The question arises what arc-length to use between samples for the this parameterization. Rather than using a fixed length, our experiments have shown that resampling the signal at exactly as many points as the time parameterization consists of and subsequently normalizing the likelihood calculated from the HMMs, result in better performance than using a fixed arc-length throughout. This lessens the chance of high frequency components in forgeries, which could contain discriminative information, be lost. Furthermore, nothing prevents us from matching signals with different arc-length interspacings to a single model as the more important issue is that the full duration of the signal be preserved and the transformation action be repeatable i.e. are done according to a fixed recipe.

Figure 4.5 shows the alignment of the path tangent for the training set of one signer under the two parametrizations. We can see how the path tangent appears slightly more stable under an arc-length parameterization. The arc-length parameterization tends to fan the signals out



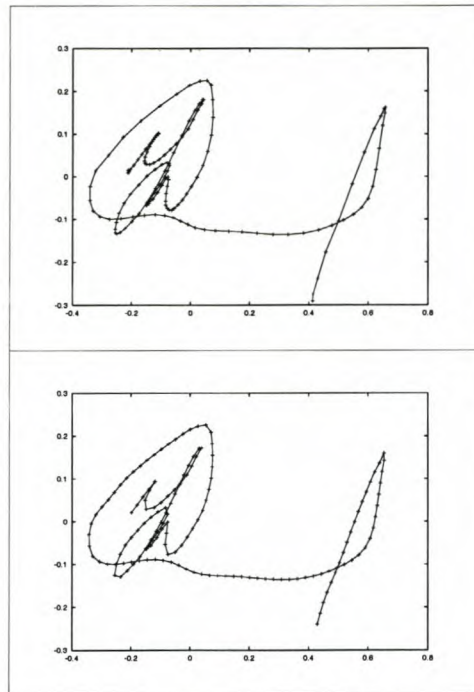


Figure 4.4: **Arc-length Parameterization:** A signature sampled on equal time and equal arc-length.

at regions which otherwise appear spiked. It is our experience that this results in slightly better alignments within the HMM, for some components more than other.

## 4.5 Hidden Markov Models

After data acquisition and preprocessing are completed, the actual modelling phase commences. As stated earlier, we need to decide upon the model state and observation sequence semantics prior to creating a HMM. If any data processing is needed to convert the data to conform to these semantics, it is performed now. Once done, a model is created and initialized from this data. This often involves inspection of the training set in a specific way to extract initial values for the model. Initialization plays an important role prior to the maximum likelihood estimation training as sensible initial values will result in faster convergence to a local maxima. If model parameters are all initialized to random values, training will converge to some local maxima but the solution is likely to be far less than optimal in terms of attained performance levels i.e. FAR and FRRs.

After initialization, the Viterbi training algorithm described in section 3.4 is used to refine

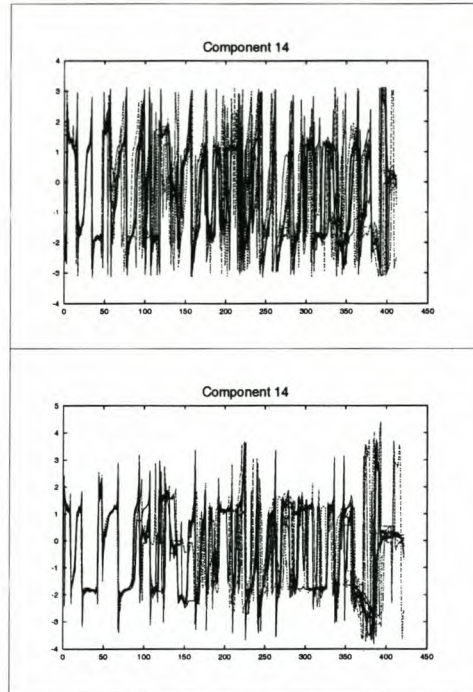


Figure 4.5: **Alignments Under Different Parametrizations:** The figure illustrates how an arc-length parameterization tends to fan out the data at regions of high density in the time parameterization of signals.

model parameters to maximize the likelihood of the training set being generated by the model. Training is terminated once the likelihood of training sequences given the current model does not increase or, increases with values less than an acceptable threshold. This is important if a system is to find its way into a production environment. Once the model is trained, the training set is presented to the trained model again, this time to determine the likelihood of each entry in the training set given the model. These likelihoods are used to determine a threshold value within the framework of some similarity measure. This threshold enables the verification of suspect signatures by comparing the likelihood of such signatures to it.

It is important to note that unlike the majority of ANNs, HMMs do not need negative examples in the training set. This means that an ASV system relying on HMMs does not have the need for forgeries to function efficiently. Forgeries are only used during the research phase to gauge the accuracy of the system.

This section proceeds to describe a number of explored model configurations in search for one to be used by the system. Different similarity measures are described in Section 4.6. The results of the experiments conducted with the models described here are presented in

Chapter 5 along with a description of their implementation in software.

#### 4.5.1 General HMM Initialization Scheme

In this study, we make exclusive use of *left-right* HMMs. *Ergodic* models have been tested in [71, 65] and underperformed compared to the results for the left-right models in the same studies (see Section 3.5 for an explanation of the two types of models). Apart from this evidence, the behaviour of ergodic models is not easily predictable, i.e. we cannot intuitively determine what set of sequences will match the model closely apart from those in the training set. This is not the case for left-right models which are sparsely connected according to easily interpretable rules. Left-right models conform more closely *only* to the modelled sequences whilst allowing for deletion and repetition of some parts as well as marginal deviation from the sequence. Ergodic models may, however, match sequences not initially intended which is a wanted feature if generalization is important; for signature verification, we opt for predictability. Ergodic models are better at modelling *nonstationary* processes whereas signature creation is arguably a *stationary* process for which left-right models are more suitable.

Having a model with no or little uncertainties is an important factor [51]; our first and most important evaluation of an ASV system must be from our common sense understanding of the algorithm and not from the reported results. This is due to the dubious nature of the quality of benchmark databases from which results are derived.

We have formulated a generalized method to build a left-right HMM from a given set of training sequences independent of their semantics. This method solves the problem of setting the initial values for an HMM prior to maximum likelihood training. It also helps us to decide on the number of states to use for the model. The method is described as follow:

- Dynamic time warping (DTW) (see [57, 58] for an in-depth explanation) is used to find a sequence in the set which results in a smallest total alignment cost with all the other sequences in the set. For multi-variate sequences, a designated single component is inspected for a least cost alignment.

- Each component of every sequence in the set is then aligned separately with the corresponding component of the reference sequence. The output of the DTW algorithm is a unifying time function which maps points on one sequence to corresponding points on the other sequence. A new sequence of the same length as the reference sequence is created by traversing the timing function with respect to the reference sequence and re-sampling the aligned sequence according to this. Figure 4.6 shows an example alignment of one of the components in a set.
- The alignment is now used to initialize an HMM. Whereas most artificial neural networks (ANNs) seek to find interclass boundaries, HMMs seek to position model parameters on the data [12]. This alignment of data assists us to do just that. The HMM is created with the same number of states as the length of the sequences in the alignment (which are all equal as explained). Kohonen Self-Organizing Feature Maps [38] (SOFMs) are then used to determine the centers of a prespecified number of kernels for the observation density functions per state. If at a certain cross-section through the set of aligned sequences, we observe a large amount of variance due to signal instability in this region, then the SOFMs will spread the kernels more to cover the entire cross-section.

During DTW, a transition histogram matrix is maintained by inspecting the timing function of the alignments. The actions taken to find a least cost path through the alignment cost matrix for each individual sequence with the reference sequence, are recorded in the histogram. This histogram is then used to initialize the state transitions of the HMM after normalizing it so entries are probabilities.

At this stage, we can impose constraints on the possible transitions by limiting the distance of transitions to states to the right from each individual state. Limiting transitions are important for restricting horizontal deviation of a signal from the reference signal. The further a signal can make a transition to another state, the more it is allowed not to conform to the reference signal by not containing parts of it and still being able to match the model to a certain extent. Such restrictions are also important for governing the size of a model; the more transitions are allowed, the more parameters it contains and the more computational intense it becomes to match signals to the model.

We can also enforce a rule which requires states that are skipped to be assigned a lower

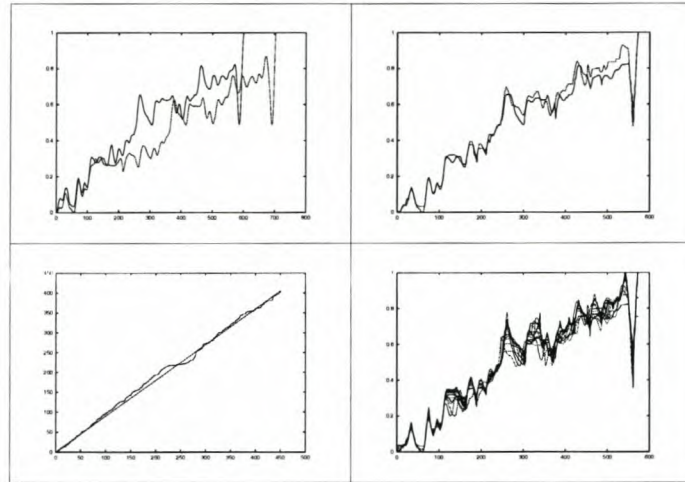


Figure 4.6: **Dynamic Time Warping:** DTW performs a non-linear alignment between two sequences. The figure shows two raw sequences (top-left) aligned by DTW (top-right) with timing function (bottom-left) and an entire set aligned to reference (bottom-right).

limit probability. This leaves the ultimate decision on its reachability from a specific state to re-estimation training. The timing functions are also used to set initial values for the maximum allowed duration counts as described in Section 3.6.

Apart from allowing us to dictate the adjustment window size i.e. how far the non-linear timing function may deviate from the linear timing function and to impose global slope constraints on the timing function, DTW is fairly restrictive as far as controllable parameters are concerned. When we look at the different parameters controllable in an HMM, we can clearly see the advantages HMMs have over DTW. The Viterbi algorithm perform the same task as DTW, i.e. it finds an alignment of the observation sequence with the model states; in our scheme, it does in fact correspond closely to the reference sequence because we use left-right models which can be seen as a memory of the sequences along with the observed horizontal and vertical deviations (see Section 3.5).

An HMM enables localized control of the amount of allowed time warping through the transition probabilities and explicit duration modelling extensions. This can be seen as *horizontal* control of the alignment: we restrict the amount of stretching adaptively for each local section

of the reference sequence according to the timing variances detected during alignment and refined during re-estimation training. Also, whereas DTW allows repetitions during alignment, HMMs allow deletions as well. This enables a Viterbi alignment to skip over a state thereby ignoring certain parts of the reference sequence in favour of other parts which matches the input sequence more closely.

*Vertical* control is achieved through the average and deviation parameters of the observation symbol density functions at each state. This provides us with the ability to adapt the variance at different points along the sequence of averages according to the variances revealed in the training set. This control is essential for building an adaptable verification system as the consistency of signatures can greatly vary among different signers as the alignment in Figure 4.7 illustrates. The fact that the density functions can be described by more than one kernel further enhances the control provided.

Figure 4.8 shows a sequence modelled by a left-right HMM initialized in this way along with the alignment from which it was initialized. The figure also shows the boundaries of the standard deviation from the centers

The main purpose of this scheme is to allow us to separate the task of initializing HMMs from the task of defining feature sequences representing the signatures. Within this framework, a state at any specific position in the model acts as an envelope for memorizing the character of the signal at similar offsets. If any more sophisticated meaning is to be coupled to a state, one needs to look at the semantics of the input sequences as there exist a one-to-one relationship for left-right models. This is generally not the case for ergodic models where state semantics and sequence semantics are often not so closely tied. For left-right models, a labelling of the sequence values (matching observation symbols to states) is implicit by the position of an observation in the sequence. This scheme would be harder to implement for ergodic models as such an implicit labelling cannot be assumed here. In [48] and [65], a simple ergodic model is built where a state groups observations called strokes which were made in the same general direction. Here, states are reused meaning that once they are exited, they can be revisited somewhere further down the sequence.

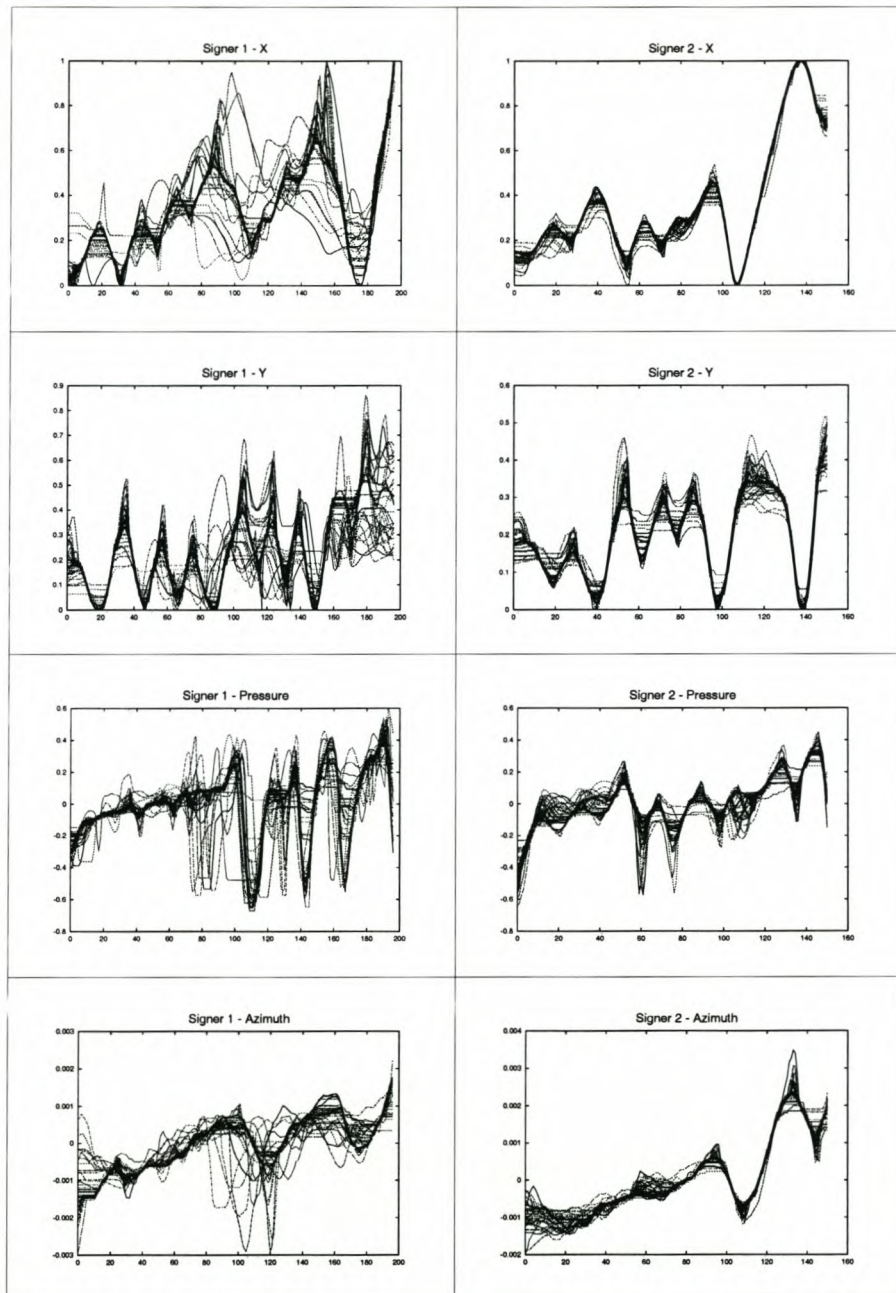


Figure 4.7: **Signer Consistency:** The sampled signature components for an inconsistent signer and a consistent one.

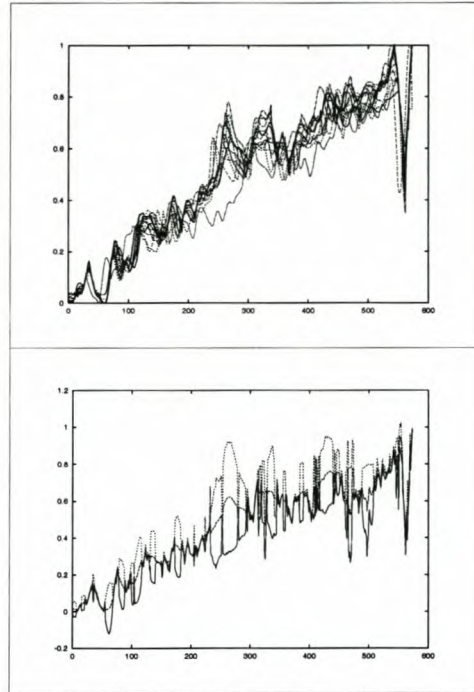


Figure 4.8: **HMM Sequence Margin:** An alignment with the initialized HMM. The vertical control is shown as the standard deviation corridor round the mean of the state observation densities.

#### 4.5.2 Reestimation Training

Once we have an initialized model by the scheme described in the previous section, Viterbi training is used to refine the model parameters. Each training pattern used during the alignment, is presented to the algorithm as part of a training batch.

We impose a restriction on the allowed terminal state (as described in Section 3.4) as typically being no further from the rightmost state than 10% of the total number of states. In doing so, we force input sequences to comply with more or less the entire model and prohibit partial matches. This means that a paraphrased version of a signature will not match the model as it will most probably require termination somewhere in the first half of the model. This terminal restriction is important and not generally used in other applications of HMMs, e.g. speech recognition where there is generally no intent to fool the system. We govern the allowed transition distance from a state to other states to the right of it to typically 5% of the total number of states. Both the terminal fraction and transition fraction depend on the size of the model and stability of the feature signals and is established empirically for each model type.

Following then, is a description of the different models explored by this study.



### 4.5.3 Single-variate Signal Memory

When building a left-right model for ASV, we seek to find stable signals to serve as signature representations i.e. signals which exhibit minimal *horizontal* and *vertical* variance amongst different authentic samples. If we can succeed in finding such a signal without compromising the signature information content, its model is likely to maximize the inter-class distance between likelihoods for authentic signatures and forgeries. The less variance a model exhibits, the less the chance of a forgery being able to exploit this variance to masquerade as authentic. Of course, this holds true only if the signature is all but trivial to forge in which case these signals, although stable, are easy to reproduce. In [15], a complexity measure for signatures is formulated which can serve to determine, beforehand, how easily a signature is expected to be forged.

If a signature is not signed consistently such that the sampled signals exhibit this stability property, one can turn to derived signals which might succeed in discarding noise without losing too much information in the process. Unfortunately, deriving such signal representations proves to be non-trivial. We now describe a number of signals derived from the original sampled signature components to be used in our experiments. For the sake of interest, Figure 4.10 provide sample plots of some of these for the signature of Figure 4.2.

#### Velocity

Velocity is regarded as the most important discriminating feature of signatures [55] (with the odd exception [51]). It is argued that a forger may succeed at duplicating the shape of a signature but will have difficulty in doing so at the same tempo as the original signer. To quote [32]:

*“An experienced static signature forger has little concern for the dynamic aspect of the signature and is likely to be foiled when his speed function is compared to that of the owner of the signature. This is especially true when he has to learn to reproduce exotic flourishes, often illegible, but consistent in the victim’s signature.”*

Velocity can be calculated from the positional signals as (see [41])

$$|V(n)| = \frac{\sqrt{\Delta x(n)^2 + \Delta y(n)^2}}{\Delta t(n)}$$

where  $\Delta f(n) = f(n+1) - f(n)$ . Alternatively, the first derivatives,  $V_x(t) = D_t x(t)$  and  $V_y(t) = D_t y(t)$ , of functions fitted to the  $x$  and  $y$ -signals can be taken and used to compute velocity as

$$V(t) = \sqrt{V_x(t)^2 + V_y(t)^2}$$

As mentioned earlier, we fit cubic smoothing B-splines to the measured signals which give access to derivatives<sup>3</sup>. We prefer to stick to the term 'velocity' as this is how this feature is commonly coined in the literature. Physicists, however, might point out that it should in fact be referred to as 'speed'. Velocity is the rate of positional change of an object *in a certain direction* whereas speed is the magnitude of such a velocity vector. Speed by itself cannot represent a signature unambiguously. Even though it is highly unlikely for a forger to recreate a velocity profile in a random way, predictable behaviour of a system is an important consideration. Note that we can easily create an actual velocity signal by creating two-dimensional observations of speed and direction. Speed is in itself rotational invariant which is an important attribute if no normalization of signatures is performed. Figure 4.10(a) shows a velocity profile.

### Path Tangent

The path tangent is the missing directional component of velocity. In the previous section, we have shown how  $V_x$  and  $V_y$  are calculated. The path tangent is related to these values by

$$T_\theta = \tan^{-1} \frac{V_y}{V_x}$$

Figure 4.10(b) shows a path tangent profile.

### Acceleration

Systems based on instrumented pens measure accelerations involved during the signing process directly. When acquiring signatures through tablets we generally need to calculate the

<sup>3</sup>The literature agrees that cubic smoothing B-splines are an appropriate choice [37, 33, 32].

acceleration profiles from the positional signals as a post acquisition step. We achieve this by taking the second derivative of the splines fitted to the sampled  $x$  and  $y$ -signals. The total acceleration is related to the axial accelerations by

$$A(t) = \sqrt{A_x(t)^2 + A_y(t)^2}$$

It should be noted that taking second derivatives is a process known to be numerically unstable [33, 51]. We will however still explore its discriminating ability as a feature signal. Figure 4.10(c) shows an acceleration profile.

### Tangential and Centripetal Acceleration

The total acceleration driving the signing process can be dissected in various ways. In the previous section, we have seen how the total acceleration can be derived from the accelerations in each of the two tablet surface axes. Another way of breaking up the total acceleration is to look at the tangential and centripetal acceleration of the pen tip.

We will explain tangential and centripetal acceleration by way of an example. Figure 4.9 shows a circle being drawn by a compass. When the compass rotates for a time  $t$ , the tip draws an arc which sweeps out an angle  $\theta$ . The distance the compass tip travels is denoted by  $r\theta$  where  $r$  is the radius of the circle and  $\theta$  measured in radians. The magnitude of the tangential velocity then is simply  $V_T = \frac{r\theta}{t}$ . Another quantity, called the angular speed, is defined as  $\frac{\theta}{t}$ . We can imagine that the magnitude of tangential velocity will not remain constant during the drawing of the circle i.e. *nonuniform circular motion*. Furthermore, the direction component of the tangential velocity is also changing continuously as the compass creates the circle. Two acceleration components are simultaneously involved to facilitate this. The tangential acceleration  $A_T$  causes the change in magnitude and is related to the tangential speed  $V_T$  by

$$A_T = \frac{V_T - V_{T0}}{t}$$

Centripetal acceleration is responsible for the change in direction of tangential velocity. It is expressed as

$$A_C = \frac{V_T^2}{r} = \frac{(r\theta)^2}{rt} = r\theta^2$$

and points to the center of the circle i.e. the tip of the static compass leg.

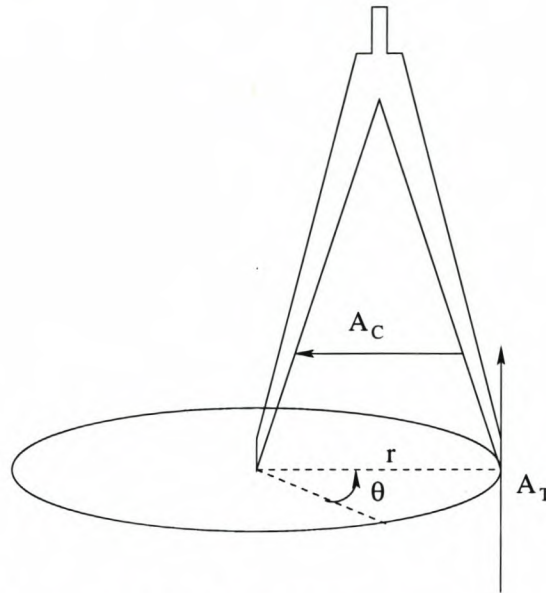


Figure 4.9: **Tangential/Angular/Centripetal Acceleration:** The different accelerations involved in a revolving object.

We calculate these quantities for a signature in a slightly different way [52]. The tangential and centripetal accelerations are respectively calculated as

$$A_T = \frac{V_x A_x + V_y A_y}{V}$$

and

$$A_C = \frac{V_x A_y - V_y A_x}{V}$$

Total acceleration is related to these acceleration components by

$$A(t) = \sqrt{A_T(t)^2 + A_C(t)^2}$$

Figure 4.10(d) and (e) shows tangential and centripetal acceleration profiles.

### Pressure Derivative

It is easy to imagine that users might not always sign with the same average pressure applied to the pen tip possibly due to mood changes. Inspection of signatures from our experiments supports this claim. We would hope that even though the average pressure may vary between exemplars, the overall profile will remain consistent, i.e. the pressure signal is merely shifted vertically between different signatures. Other studies [55] have found this to be true for some

writers; however, there are writers who do not reproduce pressure patterns with a high degree of consistency and some for which there is no consistency at all. This does raise questions about the suitability of pressure as a distinguishing feature for signatures.

To reduce the effect of vertical shift of the pressure signal, we calculate two signals from the sampled pressure signal. The first removes the DC component from the signal by subtracting the average pressure over the entire signature from the signal. The second method fits a spline to the pressure signal and calculates the first derivative from this function. Unfortunately, the derivative has the negative side effect of destroying the pen-up/down information. Figure 4.10(f) shows a pressure derivative profile.

### Pen Tilt Azimuth

The tablet reports the pen tilt as the two angles the pen makes with the  $x$  and  $y$ -axes of the tablet surface. It is not clear whether the absolute tilt angles are stable enough to serve as a distinguishing feature for signatures. The pen tilt azimuth, depicted by  $\varphi$  in Figure 4.3, is deduced from the sampled tilt angles. This quantity has the added advantage that it is a rotational invariant and rather depends on the way a writer generally holds his/her pen. Figure 4.10(g) shows a pen tilt azimuth profile.

### Line Thickness

Here, we introduce an imaginative quantity which could be seen as a measure of the thickness of the line constituting a signature. This quantity combines the pen tilt azimuth, the pen tip pressure, the path tangent and the speed at which the line is drawn. The formula makes the line thicker as the angle between the current stroke direction and the direction the pen points at (when viewed from directly above) increases, up to a perpendicular angle. Furthermore, the more upright the pen is held the less pen tip contact with the surface resulting in a thinner line. The formula also argues that the slower the stroke is made, the more time for ink to flow from the reservoir. The amount of ink is governed at a certain amount to prevent infinite

thickness when the velocity is zero. The formula is defined as

$$LT = \frac{\frac{|180^\circ - (PD \angle SD)|}{90^\circ} + P}{1 + e^{-V}}$$

where

- LT is the Line Thickness,
- PD is the Pen Direction,
- SD is the Stroke Direction,
- P is the Pressure and
- V is the velocity.

Figure 4.10(h) shows a line-thickness profile. These calculations were based on our observations writing with a soft-tip ink pen. Different formulae can be derived for different types of pens but ultimately one would hope to simulate some form of calligraphic pen tip which shows fluctuation in line thickness depending on the way the pen is controlled.

For each of the original sampled signals and these derived signals, we build a single-variate HMM with no signal compression. By uncompressed we mean that there was no segmentation of the signal where each segment is represented by a smaller set of values (see Section 4.5.6). We use the general initialization method described in Section 4.5.1 to align the original and derived signals as is. This results in fairly large HMMs in terms of the number of states.

Such an experiment enables us to measure the individual discriminating ability of each feature signal without having to fear that the results were influenced by a flawed data reduction scheme. As each signal possesses differing ability to distinguish authentic signatures from forgeries, we can use optimal threshold values for each model separately. From the error rates for each feature signal, we can now determine weights by which the features are regarded in a unified decision function. Apart from these considerations, we have found that the timing information for the different features does not necessarily correspond closely. This implies that the timing function which aligns the velocity signals for instance does not necessarily align the pressure profiles optimally. Figure 4.11 shows two instances of a component aligned

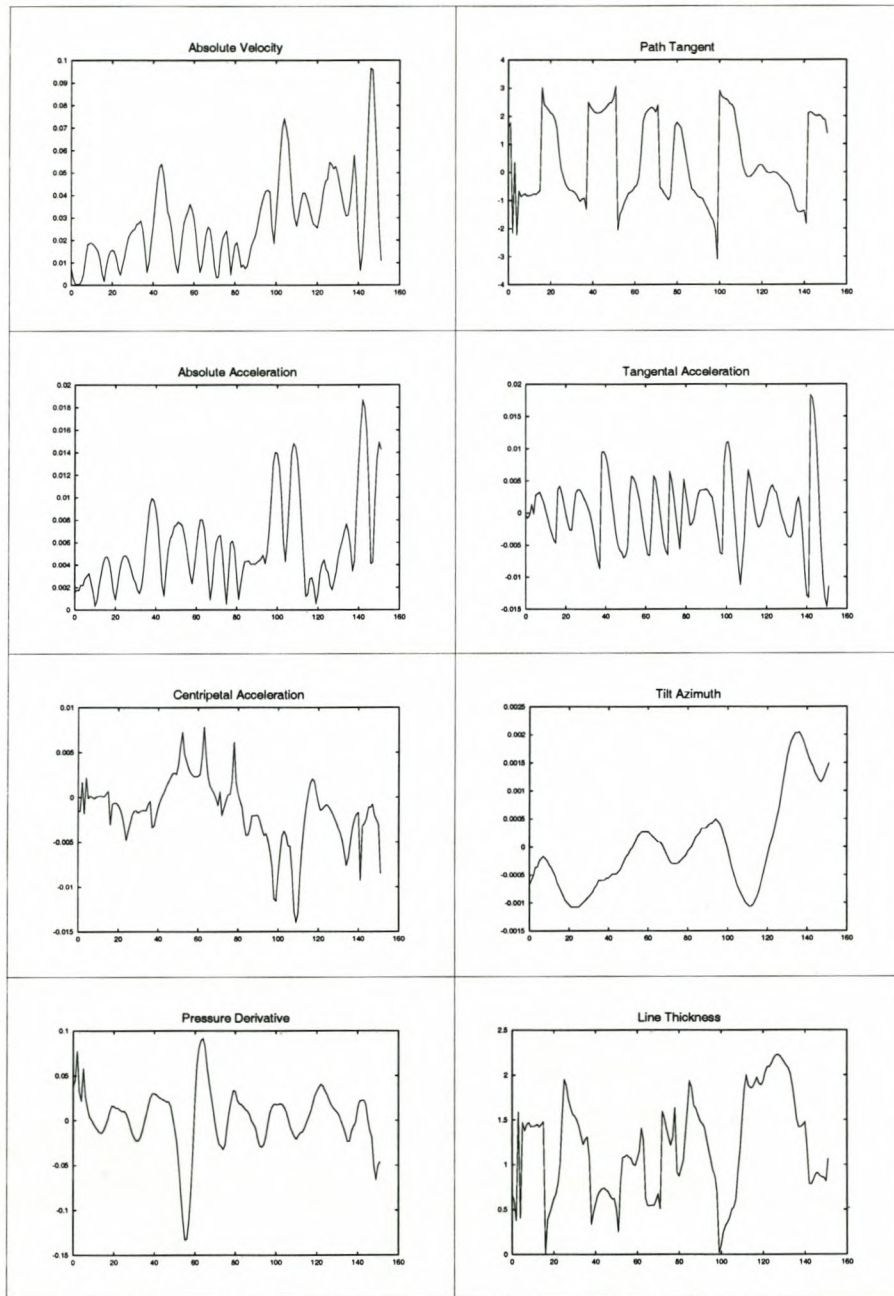


Figure 4.10: **Derived Feature Signals for Signature of Figure 4.2:** (a) Absolute Velocity, (b) Path Tangent, (c) Absolute Acceleration, (d) Tangential Acceleration, (e) Centripetal Acceleration, (f) Tilt Azimuth, (g) Pressure Derivative, (h) Line Thickness.

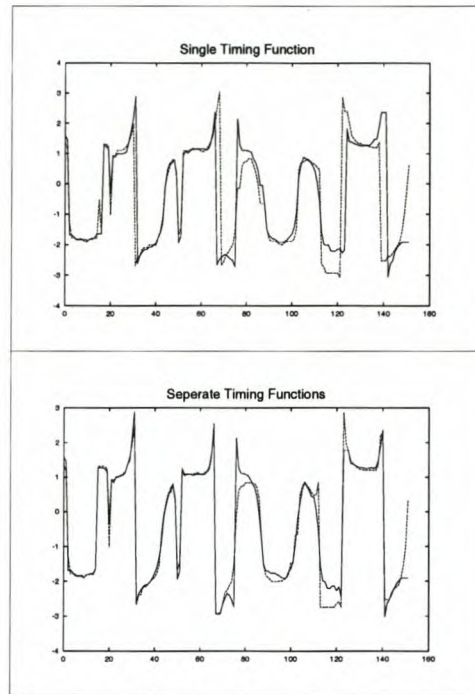


Figure 4.11: **Separate and Unified Alignment:** This figure illustrates the variations in timing information between different components in a single signature.

by the timing function of another component and aligned by their own timing information. This suggests less correlation between the timing information of individual feature signals than what might be assumed at first. Having separate HMMs for each feature means that we can model each feature's timing information separately through the transition probabilities in the models.

This experiment is conducted on both a time parameterized and arc-length parameterized version of the signature database. In the latter case, the horizontal control of a HMM model shapes variations in the signature rather than timing variations. The tablet from which the signature database was built has the ability to track the pen up to a certain height above the surface. It is thus possible to have sections of zero pressure trajectories as part of the signatures. We inspect both versions of the database where such sections have been left intact and where each has been replaced by a single point at the middle of the line connecting the two adjacent pen-down sections. The results of this experiment serve as an indication of the (in)stability of such pen-up sections in signatures. Figure 4.12 shows the pen-up segments plotted as dotted lines for a number of exemplars of a single signer. For this specific signer, pen-up information appears fairly unstable. The results in the next chapter provide further



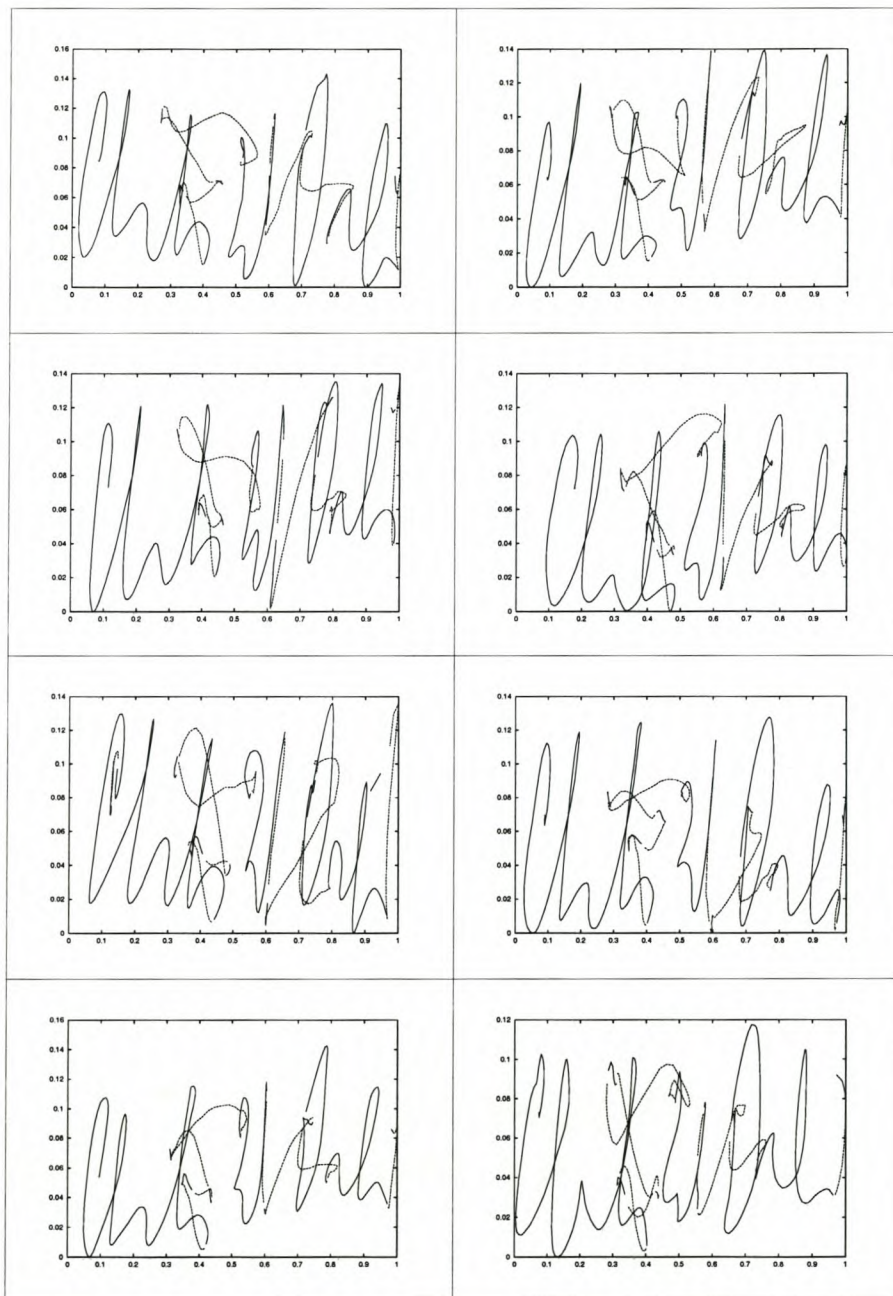


Figure 4.12: **Pen-up Information:** The figure shows the relative unstable pen-up sections of a signature.

insight into the stability or not of this information. On average, this omission reduces the signature data by 20%.

As this experiment deals with large models, we investigate a variation thereof where a model is built from a reduced version of the signatures. We dictate the reduction ratio and resample a signature by representing a number of consecutive samples by a weighted average where the sample at the centre contributes the most and then decreases linearly to each side up to a designated distance. These sparsely sampled signatures are now used to initialize a model exactly as we do with unreduced signals. After initialization, HMM re-estimation training commences with the unreduced signatures. In order to use this model for matches against unreduced signatures, we increase the allowed state durations by multiplying them by the reduction ratios. If transition fractions similar to the unreduced models are to be imposed, a smaller transition fraction, scaled by the inverse reduction ratio, has to be used.

A shortcoming of the filter approach used here becomes apparent when a high reduction ratio is imposed on a signal with steep slopes between consecutive inflection points. The path tangent is an example of such a signal. The effect of a moderate reduction ratio is however not so serious and can assist in reducing model size without too much of a performance penalty.

Section 4.6 describes how the likelihood values computed for each of the individual models are combined to reach a decision on the authenticity of a signature. Results for this experiment and those described below, are presented in Chapter 5.

#### 4.5.4 Multi-variate Signal Memory

Unlike the single-variate case where we create a model per feature signal, the multi-variate signal memory uses a single model with multi-variate observation density functions to combine the feature signals. This means that a single horizontal timing function is used to align the different features within the verifier. We can use the results from the single-variate experiment to combine *good* feature signals into multi-variate models in search for a model with a large inter-class distance between authentic signatures and forgeries.

For the purpose of illustration, one such multi-variate HMM can be constructed for the two-dimensional sequence of  $(x, y)$  values. This enables us to model the shape variations



Figure 4.13: **A Two-dimensional HMM modelling  $(x, y)$  signals:** The HMM positions observation density functions along the signature trajectory widening and narrowing them as needed.

of a signature. Figure 4.13 shows a contrived example of a scribble with two-dimensional density functions positioned along the trajectory as such a model would do. The most basic requirement for most signatures is that they need to appear more or less the same [51]. Such a model positions two-dimensional density functions along the entire length of the signature automatically widening or narrowing as the signature consistency varies in each local region. We can expect that the timing information for these two signals are closely correlated which warrants the use of a single timing function to align the signals without a significant loss in performance.

As stated in the previous section, experiments with the DTW-HMM initialization scheme lead us to believe that a single timing function does not necessarily align all the features of a signature. However, if the timing information of two (or more) features are closely correlated, a multi-variate model reduces the total model size by using only a single set of transition probabilities for all the constituent features. Comparison of the results for multi-variate models and that of a combination of corresponding single-variate models, gives insight into the importance of separate timing functions.

Furthermore, the single-variate models enable us to standardize the individual log-likelihoods of the components separately as they might very well come from different underlying distributions (see Section 4.6). For the purpose of this study, we will accept the evidence provided by the comparative results in Chapter 5 without resorting to a formal analysis of the advantages and disadvantages of each approach.

#### 4.5.5 Randomly Initialized Model

In order for us to evaluate the claim that random initialization of an HMM results in a sub-optimal model being discovered by re-estimation training, we have created such an HMM. In this model, the parameters are to be discovered entirely by maximum likelihood estimation training. The only information extracted from the training set to be used for initialization, is the average sequence length which determines the number of states in the model. The observation density functions, start and transition probabilities and duration counts are all initialized to random values within the framework of a left-right model. The training set is then presented to this model for re-estimation training.

As input sequences, we used the uncompressed original and derived signals parameterized by time. The results from this experiment can be directly compared to the the results of the models in the previous section for insight into the benefits of sensible initialization of a model prior to re-estimation training.

#### 4.5.6 Statistical Segment Description Models

A problem with the signal memory models described in the previous sections is that they require a significant amount of states which implies more computational overhead. A solution to this problem is to use some segmentation scheme to break a signature into smaller segments. A segment is then represented by calculating a set of descriptors for it. Here, a descriptor is a numerical value which records some piece of information about the segment. The signal is thereby reduced in length, however, the dimension of the data is increased by a multiple of the number of descriptors. We achieve lossy compression if the amount of horizontal reduction

outweighs the amount of vertical increase.<sup>4</sup> Naturally, we investigate schemes which result in a small set of descriptors whilst minimizing the loss of relevant information. It is important that the descriptive values be conceptually likely to form clusters with normal distributions for the set of training signatures as this will validate our use of these distributions in the HMMs when modelling such compressed signals. The segmentation points must correlate amongst different samples of the same signature, i.e. result in visually similar parts being isolated by the segmentation. This will ensure that descriptors will be calculated for similar partial signals. Such descriptors can then be modelled by an HMM and later compared to signatures segmented by the same criteria.

The question now arises as to what constitutes a segment. We choose to break a signature at points where the velocity in the  $y$  direction,  $V_y$ , becomes zero. This approach is not uncommon for segment-based signature models and is also used in [23] as it is argued that it results in a segmentation independent of the signature size. In [32], a signature is segmented at places where the absolute velocity  $V$  drops below 15% of the average velocity. Such segments are referred to as letters even though they may not in fact correspond to alphabetical letters. A similar approach is used in [55] which segments signatures at points in the trajectory which simultaneously correspond to discontinuities in angular velocity and null curvilinear velocity. [37] breaks a signature into segments of approximate equal spatial length and refers to such segments as strokes. [54] divides a signature into pseudo stationary sections. Zero absolute velocity is used in [48] as it is argued that this results in a basic stroke with minor activity at the beginning and end of the segment. What most of these segmentation schemes have in common is that they seek to break a signature into natural units of writing. Such a unit must exhibit relatively little oscillatory behaviour among the features used to describe a signature. We can say that we seek units with as little as possible entropy across the signature feature space. Such units are well suited to a lossy compression scheme. Breaking at points of high curvature is generally a good approach to finding such units. As mentioned previously though, calculating curvature requires second derivatives which is unstable and stability is very important when segmenting a signature.

Figure 4.14 shows a signature segmented on  $V_y$  inversions and  $V$  below different percentages

---

<sup>4</sup>The term *lossy* refers to the fact that the original signal cannot be exactly recalled from the compressed version.

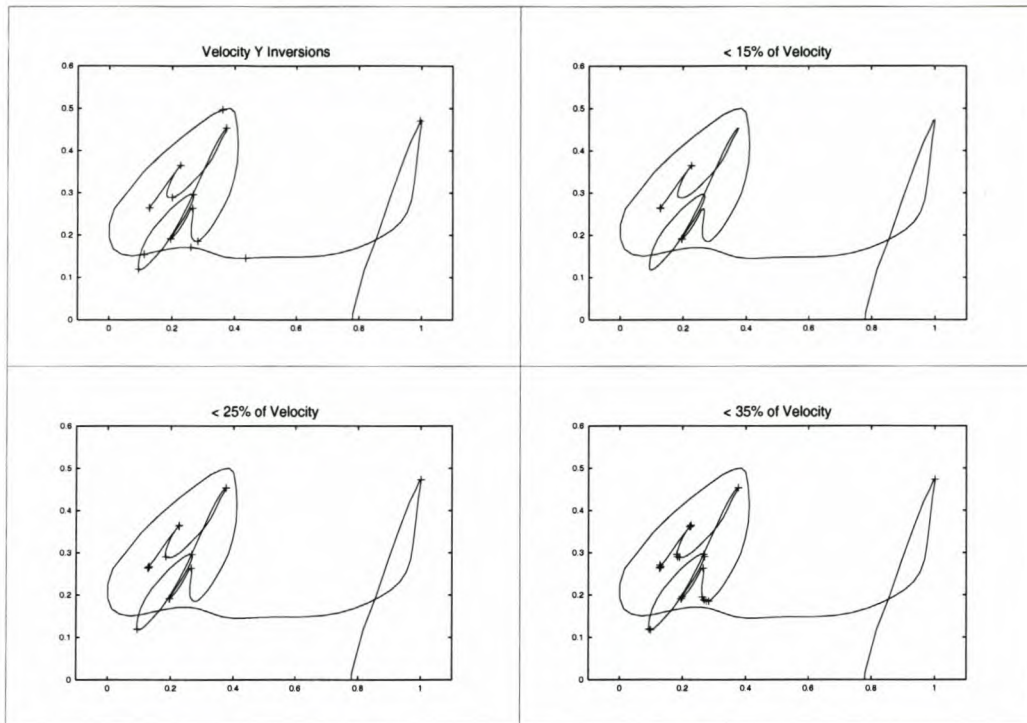


Figure 4.14: **Segmentation of a signature according to different strategies:** A signature is segmented according to inversions in the  $y$  velocity and when the absolute velocity drops below a percentage of the average velocity.

of the average  $V$ . We see that a 15%  $V$  threshold gives a poor segmentation whereas the  $V_y$  approach does fairly well. It finds all points of high curvature (and unfortunately a bit more). Perhaps for this particular signature, the 35%  $V$  threshold achieves the best segmentation. It is however difficult to know what the threshold should be as it differs between signatures. We therefore stick to the  $V_y$  inversions and bear with the fact that it will sometimes create more segments than needed. This can be compensated for by allowing a left-right HMM to skip over the states representing such *abnormal* segments.

[51] argues that segmentation based approaches are in general not robust and deteriorate rapidly in the presence of segmentation errors which are bound to occur at some stage. We have found that a more consistent  $V_y$  inversion segmentation is achieved by applying a relatively wide low-pass filter to the  $V_y$  signal before segmentation. To some degree, this prevents noise in the signal from resulting in inconsistent segmentations. These filtered signals are only used to find segmentation points to segment the original signal.

As reported earlier, [32] attempts to minimize segmentation errors by increasing the  $V$ -threshold percentage if there is an inflection in the velocity profile below the current threshold.

This increase is, however, governed to a maximum of 18% which could prove inadequate as already illustrated.

Once we have a segmentation, we need to calculate descriptors to describe the segments. For this model, a segment is represented by the descriptor five-tuple

- the gradient of a regression line fitted to the signal,
- the average value of the signal in the segment,
- the variance of the signal around the regression line,
- linear correlation coefficient for the signal in this segment and
- time spent in segment as a fraction of the total time.

These descriptors seek to fit a line to the partial signal in the segment. The average positions the line in a vertical reference frame. The variance captures the amount of deviation from this line and the correlation coefficient measures the degree of linearity of the signal in the segment. The time fraction serves a synchronization role when the HMM is initialized from the segmented data and thereafter during Viterbi alignment.

The descriptors do not carry enough information to reconstruct the original signal but do capture important attributes about the authentic signatures which can be modelled by a HMM and used for verification. In this experiment, we create a five-dimensional model for each original feature signal and combine the output of these multi-variate models, as we do with that of multiple single-variate models, to reach a conclusion on authenticity of a signature. This scheme reduces the size of the data on average by 75%.

#### 4.5.7 Specialized Segment Description Models

In this experiment, we continue with the same segmentation scheme as described in the previous section. However, we substitute the somewhat generic statistical descriptors for a specialized description scheme which seeks to represent all the facets of a signature in a segment by using our full knowledge of the signing process. A similar approach is described in [23].

Whereas Section 4.5.6 replaces segments of each feature with  $N$  representative values resulting in a  $N$ -fold increase in the dimension of the signature data, this method replaces all the original and derived feature signals at once with a fixed number of values. These values are the:

1. line starting angle of the segment,
2. angle of the line connecting the start and end of the segment,
3. connection distance (deduced from the size normalized signature),
4. surface area between the convex/concave hull and the connecting line,
5. line ending angle of the segment,
6. the pressure range within the segment,
7. the average pressure within the segment,
8. the gradient of a least squares line fitted to the pressure signifying the pressure trend,
9.  $\Delta_{Longitude}\Delta_{Latitude}$ ,
10. average longitude,
11. average latitude,
12. maximum velocity in the vertical direction,
13. maximum acceleration in the vertical direction,
14. maximum deceleration in the vertical direction,
15. maximum line-thickness,
16. minimum line-thickness,
17. arc-length of segment as fraction of total signature arc-length,
18. duration of segment as fraction of total signature duration,
19. center of gravity  $x$ ,



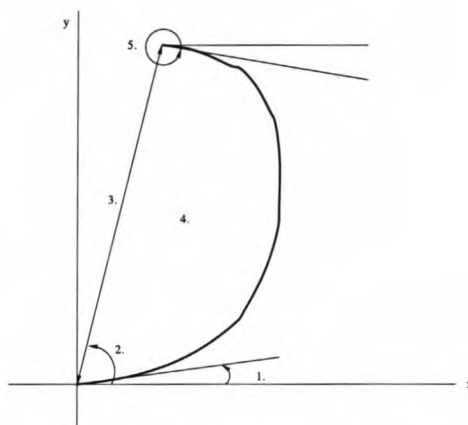


Figure 4.15: **Visuals Descriptors:** This figure illustrates five of the descriptors listed in the specialized segment description model for capturing visual information.

20. center of gravity  $y$ ,
21. distance between adjacent centers of gravity and
22. time between center of gravities as fraction of total time.

Figure 4.15 illustrates the first five descriptors listed above. Descriptors one to five seek to capture the visual qualities of a segment succinctly with as few as possible values. Strictly speaking, we could omit descriptor three (connection distance) as the other four descriptors allow for an unambiguous derivation of this value; we include it to compensate for the fact that the surface area is an approximation. Descriptors six to eleven seek to capture the invisible but directly measurable pen pressure and inclination. Here, descriptor nine seeks to measure the amount of swivel the pen underwent while the segment was written instead of including the absolute ranges of movement. Descriptors twelve to fourteen capture kinematics of the pen tip during the writing of the segment. It would not be useful in every instance to calculate descriptors for absolute velocity and acceleration within a segment as nothing prevents these values from remaining constant throughout the segment. Instead, we measure the kinematics only in the vertical direction as these are guaranteed to change seeing that our segmentation scheme relies on this fact. The minimum vertical velocity can be expected always to be close to zero (at the start and end of a segment). Therefore we capture only the maximum vertical velocity and also its rate of change as it increases in the first part of the segment and decreases in the second part. Descriptors fifteen and sixteen capture the pseudo

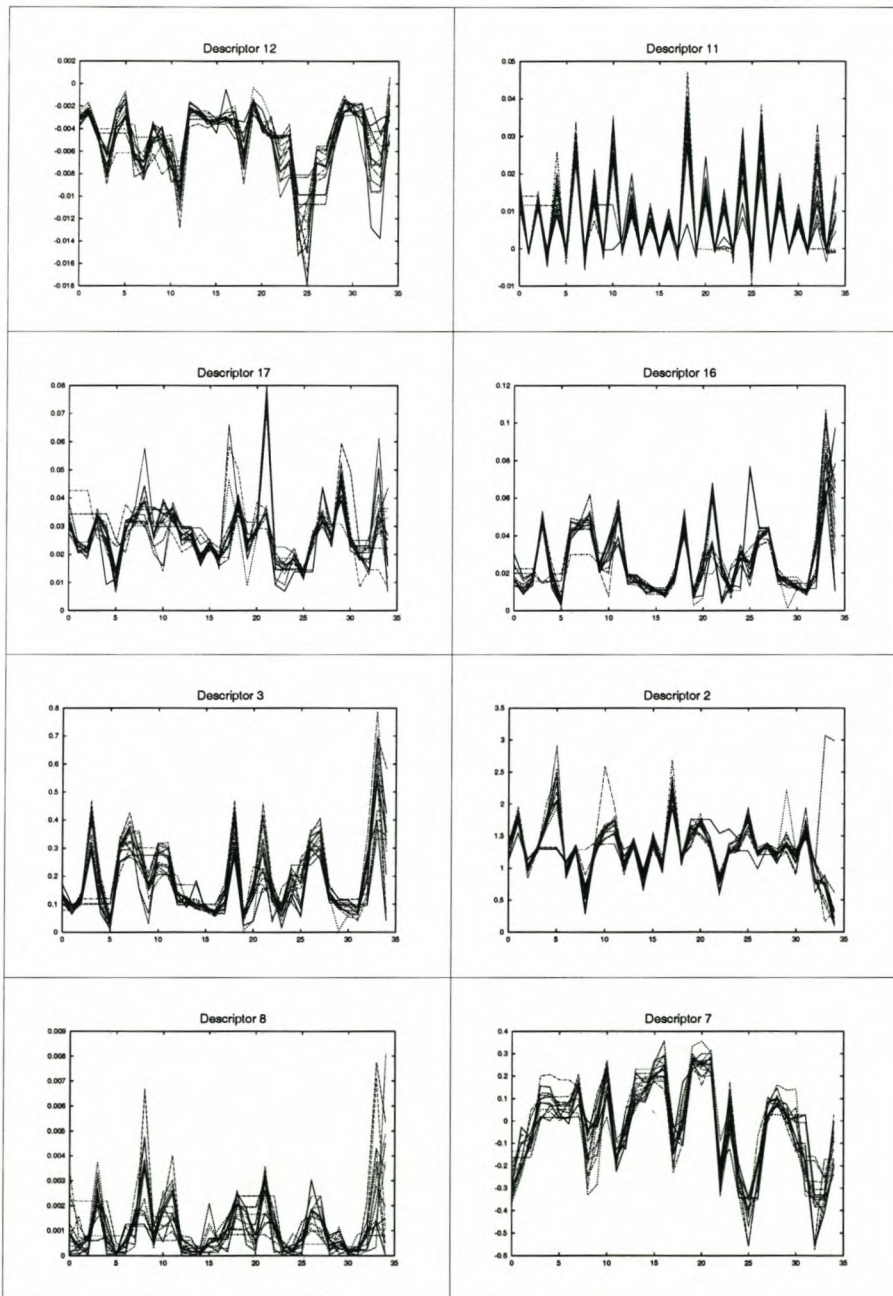


Figure 4.16: **Segment Descriptors Alignment:** An alignment of some of the specialized segment descriptors.

line-thickness characteristics for the segment. Descriptors seventeen and eighteen provide measures of a segment's creation relative to the rest of the signature. These values must be consistent for authentic signatures even when made at different overall pace. Descriptors nineteen and twenty seek to ensure that the position of a segment occurs at more or less the same place in the normalized signature. Finally, descriptors twenty-one and two tracks the overall progression of the signature. Figure 4.16 shows some of the descriptors aligned by DTW for model initialization. As can be seen, the descriptors align fairly well for this particular signer. The very first and last segments however displays some instability and the HMM needs to compensate for this by allowing large variance in the density functions for these two states.

This model seeks to capture the important attributes of a signature with a minimal amount of descriptors by using our knowledge of the problem domain. On average, the signature size is reduced by 90% resulting in a relatively compact signature model which for instance is much faster to process than the signal memory.

## 4.6 Authenticity Decision

Once the log-likelihood of a suspect signature is calculated from a HMM, we need to decide on the authenticity of the signature based on this value. As this is a likelihood value and not a probability, we cannot make assumptions about its magnitude. To make sense of it, we need to know in which region the majority of authentic signature log-likelihoods lie. This region is not the same for different users as the models for their signatures differ. In statistical terms we need to know something about the distribution of these values. The distribution refers to the concentration of the values in different regions of the definition space. If we know the distribution, we can decide upon a likelihood interval for which we are willing to accept a suspect signature as authentic given the performance constraints placed upon the ASV system.

### 4.6.1 Statistical Distributions

The distribution of a random variable in the statistical sense is often described by some mathematical function with well-defined behaviour. For a continuous random variable, the distribution may be expressed as an integral of the *probability density function—p.d.f.* For instance, the profound *central limit theorem* shows how many natural processes exhibit characteristics which conform to a *normal* distribution. Figure 4.17 shows the density function for a normal distribution with mean  $\mu = 0$  and variance  $\sigma^2 = 1$ . For real world problems, we most often do not know what the underlying population distribution is. Instead, we have a sample taken from some population from which we have to draw conclusions. If our sample is of sufficient size, it will give a good reflection of the underlying distribution. Modern statistics has produced mathematical descriptions of various frequently encountered distributions. A p.d.f. has the property that

$$\int_{-\infty}^{\infty} f_X(y)dy = 1$$

for a random variable  $X$ . If we have a mathematical model for the underlying distribution, then we can calculate the surface area of the function for any interval on the definition axis by integration. Given a certain fraction of the surface area we can thus determine an interval which will include such an area. This means that, from the density function, we can determine boundary values for the random variable which we know will include a prescribed percentage of occurrences as is depicted in Figure 4.17. The integral of a p.d.f. is called the *cumulative distribution function—c.d.f.*, and is defined as

$$\Phi(x) = \int_{-\infty}^x f(y)dy$$

Therefore, given that we know the distribution of a random variable, we can determine a threshold value  $\tau$  beyond which we are  $(1 - \Phi(\tau))\%$  sure the random variable will assume such a value. Using this approach, we can, upon encountering a value outside the range, decide that it was less likely than what we are prepared to accept for the variable to assume such value.

How does this pertain to our problem From the signature HMM of a user, we determine what the likelihood is that some test signature was generated by the same process from which the model was derived (see Section 3.3.1). As we work with continuous HMMs, the likelihood

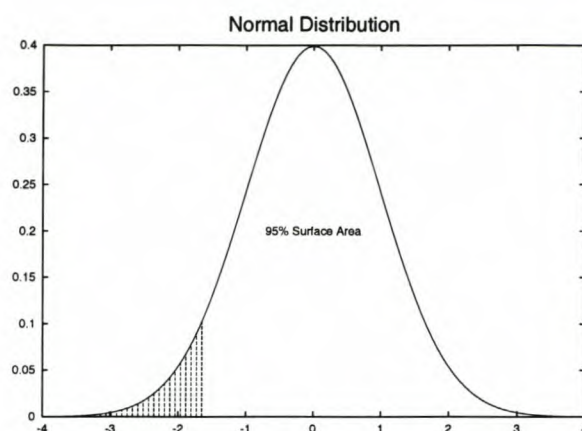


Figure 4.17: **Normal Distribution**  $n(\mu; \sigma^2)$ : The normal distribution occurs frequently in many natural phenomena. This is the observation made by the Central Limit Theorem.

function described by the model parameters is a probability density function rather than a probability function. This density function is complex and hard to analyze; finding an interval for a signature as a random variable analogous to that described in the previous paragraph is very difficult. Instead, we will inspect the distribution of likelihood values for authentic signatures in order to make decisions about the authenticity of suspect signatures.

#### 4.6.2 Signature Likelihood Values

As we work with signatures which are sequences of finite length, the likelihood value computed from a HMM has a definite ceiling. The likelihood value for a signature is also tied to the length of the signature; to minimize this dependence, we can normalize the likelihood with respect to the length by division. Furthermore, the theory of HMMs allows us to compute the likelihood of a signature on a logarithmic scale only. Recall from Chapter 3 that the reason for this is to prevent underflow of the finite arithmetic capabilities of computers. This means that, although likelihood values are positive, we end up with positive as well as negative values.

Apart from the log-likelihoods normalized with regard to signature length and the unnormalized versions, we also explore the log-likelihood distance from an optimal log-likelihood of the same length. Such an optimal log-likelihood is obtained by allowing the Viterbi-algorithm,

described in Section 3.3.2, to choose the signal values at each time instance. It does this by using the kernel parameters of those model states which maximizes a path of the prescribed length.

Figure 4.18 illustrates the distributions of authentic and forged signature likelihoods for one experiment. If a model has any power to separate authentic and forged signatures, the log-likelihood values for the two groups will be centered around different values; the forged distribution will be to the left of that of the authentic, i.e. less likely to have been generated by the model. Ultimately, we hope to find a threshold which minimizes the hatched surface area in the figure. The point of intersection of the two distributions is exactly at such a value. One must keep in mind, though, that forgeries are not readily available in a production system and finding such a threshold value is restricted to the research environment. Furthermore, the quality of forgeries casts doubt on the legitimacy of such a threshold in minimizing the real joint-error rate. Apart from this, the target deployment scenario might require for a threshold which more or less guarantees a certain FRR and not necessarily minimize the equal-error rate. We therefore will seek such an optimal threshold only in order to calibrate the system for scenarios where optimal performance is required regardless of the individual FRR/FAR. Figure 4.19 plots the FRR and FAR against the threshold value for one of the conducted experiments. The place where the two curves intersect is called the equal-error rate (EER) which is most often used to benchmark the quality of an ASV algorithm.

The study of the underlying authentic distribution remains important as this will enable us to achieve a prescribed FRR regardless of the FAR which are not dependent on the availability of forgeries. In this case, the forgeries in the benchmark database will only serve to measure the performance of the system and will have no perusal on the value of the threshold.

### 4.6.3 Threshold Calculation

As stated, the larger a sample becomes, the better it approximates the underlying true population. For a single user we typically have a relatively small sample set of signatures and therefore a small log-likelihood sample. One will not have much confidence in drawing conclusions from such a small sample as far as its statistical distribution is concerned. We therefore

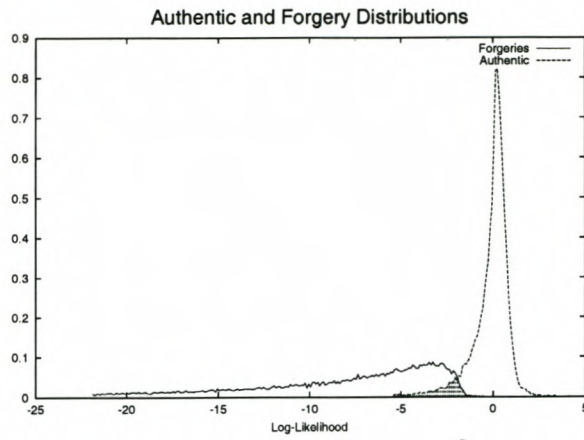


Figure 4.18: **Optimal Threshold:** The log-likelihood at which the authentic and forgery distributions intersect is a threshold which minimizes the joint FAR/FRR.

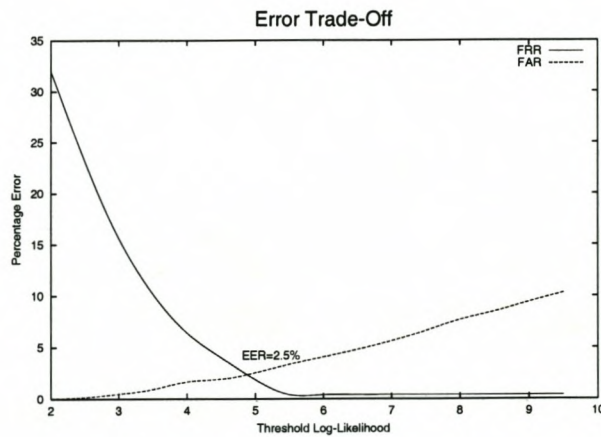


Figure 4.19: **Equal-Error Rate:** A trade-off exists between the FRR and FAR. The point of intersection of the curves are called the equal error-rate and often used to report the performance of an ASV algorithm.

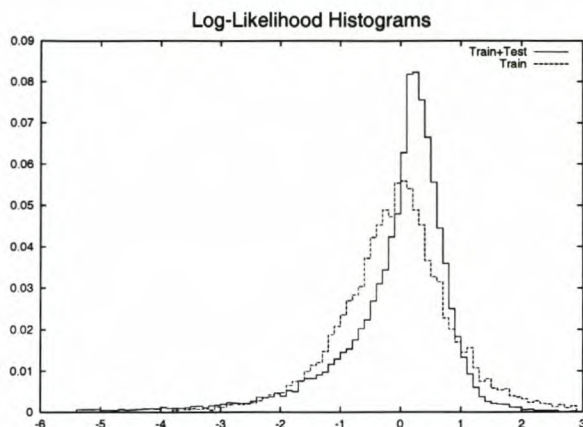


Figure 4.20: **Standardized Log-Likelihood Histograms:** The plot shows histograms for the log-likelihoods for authentic signatures according to one of the implemented HMM configurations.

transform each user's results separately to a uniform reference frame by standardization to form a new distribution for which we then have a much larger sample set. Each user's results are standardized separately by  $z = \frac{x - \bar{x}}{s_x}$  where  $\bar{x}$  is the sample mean and  $s_x$  the sample standard deviation. This is done because their individual distributions differ due to different HMM parameters. Figure 4.20 shows histograms for the standardized log-likelihoods of authentic signatures for one experiment. This gives us an idea of what the distribution of likelihood values looks like. As the plot illustrates, the standardization has the effect of shifting and scaling the distributions to conform closer to a single universal distribution of the log-likelihoods of signatures in general modelled by a HMM with certain semantics. As we now have a larger sample, we have a better chance of making a reliable fit to a theoretical distribution or approximating the underlying distribution with a discrete distribution function. This also implies that there is no need for personalized thresholds as a single global threshold can be used. The histogram for the training set alone resembles a normal distribution, whereas the histogram for both training and testing data which includes unseen authentic signatures exhibits a negative skew distribution. It is this distribution that we will explore further.

The majority of documented distributions which deal with skewness assume a positive skew profile. The histogram in Figure 4.20 shows the distribution of the joint training and testing set to be negative skew so we take the negative of the log-likelihoods and shift them by a



constant to make all the values positive. We have attempted to fit the  $\Gamma$ ,  $\chi^2$  and  $F$ -distribution to this transformed set. Maximum Likelihood Estimation is used to search for a most likely density function. The non-central  $F(11, 45, 5)$  and  $\Gamma(3.96, 4, 0)$ -distributions are the closest we get to finding a density function which fits this specific distribution. For the sake of interest, the non-central  $F(n_1, n_2, \lambda)$  p.d.f. with  $n_1$  and  $n_2$  degrees of freedom and non-centrality  $\lambda$  is defined as

$$F(n_1, n_2, \lambda) = e^{-\lambda} \sum_{i=0}^{\infty} \frac{\lambda^i x^{0.5(n_1+n_2)-1} \Gamma(\frac{2i+n_1+n_2}{2}) (\frac{n_1}{n_2})^{0.5(2i+n_1)}}{i! (1 + \frac{n_1}{n_2} x)^{0.5(2i+n_1+n_2)} \Gamma(\frac{n_2}{2}) \Gamma(\frac{2i+n_1}{2})}$$

and the  $\Gamma(\alpha, \beta, A)$  p.d.f. as

$$\Gamma(\alpha, \beta, A) = \frac{(x - A)^\alpha e^{-\frac{(x-A)}{\beta}}}{\beta^{\alpha+1} \Gamma(\alpha + 1)}$$

We use *Mathematica* to implement the MLE search as the approximation of this distribution requires intermediate calculations with larger numbers than can be represented by standard programming languages, e.g. C. Figure 4.21 shows the F-distribution p.d.f. superimposed onto the data set. It is not necessary to do a  $\chi^2$  goodness of fit test to see that the density function does not sufficiently match the distribution in order to be used for threshold calculations. In theory, a mixture of Gaussian, i.e. normal distributions, can approximate any distribution arbitrarily closely. We have resorted, though, to using the discrete probability distribution derived by a binning procedure from the sample to calculate a threshold value for a specific FRR.

As we have standardized the log-likelihoods for the different users, we have a relatively large sample which we assume to represent the underlying population fairly accurately. We create a discrete distribution function by choosing a suitable resolution for a binning procedure which is used to determine frequencies of occurrence in fixed intervals over the entire log-likelihood region. From this we create a discrete c.d.f. which maps thresholds on a regular grid to the cumulative surface area up to each threshold. We call this discrete mapping  $\bar{\Phi}(x) = y$  after its continuous counterpart. Figure 4.22 shows such a mapping for one experiment. We could fit a cubic smoothing spline to this curve for a better approximation in the limit case. Obtaining

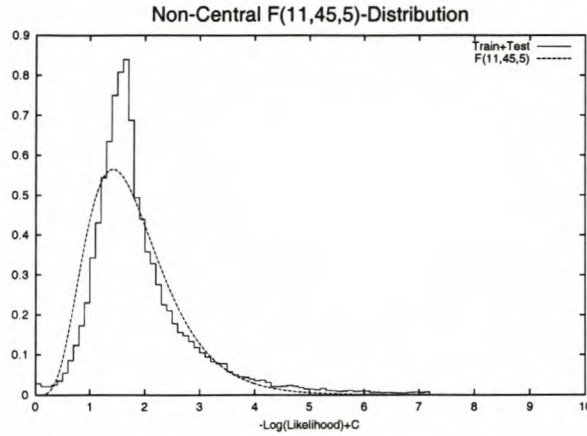


Figure 4.21: **Non-Central F(11,45,5) Superimposed on Log-Likelihood Histogram:** Maximum Likelihood Estimation is used to fit density functions to data.

a threshold for a specific FRR simply entails using the inverse  $\bar{\Phi}^{-1}(y) = x$ , interpolating where needed, to approximate the integration range which will include a prescribed surface area under the curve. For instance, if the requirement is for a FRR of 1%, set the threshold to be  $\tau = \bar{\Phi}^{-1}(0.01)$ . The discrete distribution function can be recreated in a production system from time to time as the number of authentic samples grows.

In the case where we combine the results of several models to reach a conclusion on authenticity, various approaches can be applied to reach a decision (Chapter 5 includes performance reports for the approaches described here).

The simplest approach is to conclude that a signature is a forgery if any of the individual log-likelihood values are less than their respective threshold values. In this case it becomes fairly difficult to predict the actual FRR even though we have the distribution of log-likelihoods for each individual component. This is because the actual FRR will depend on the correlation of rejections amongst components. If for instance we choose a threshold for each component which in each instance will yield a FRR of  $\approx R_c$ , an actual FRR of  $\approx [R_C]$  will be achieved only if there exists a maximum correlation amongst the set of signatures rejected by each component, i.e. any rejected signature is ruled so by all components at once. In the case of zero correlation, the actual FRR will be

$$\approx \sum_{c=1}^C R_c$$

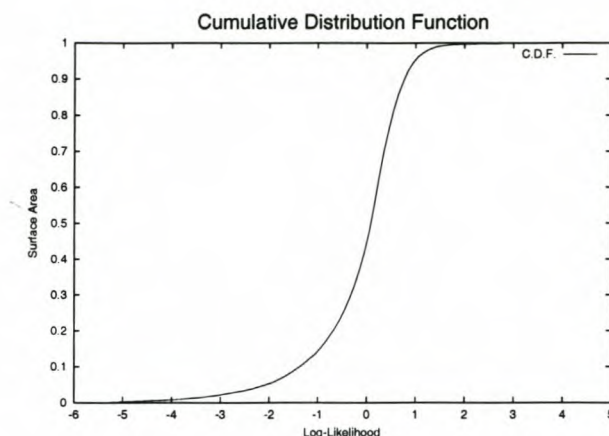


Figure 4.22: **Discrete Cumulative Distribution Function:** The inverse distribution function is used to map surface area to a threshold.

By zero correlation, we mean

$$S_i^A \cap S_j^A = \emptyset \quad \forall i \neq j | 1 \leq i, j \leq C$$

where  $S_c^A$  is the set of authentic signatures rejected by the  $c^{th}$  component. A large degree of correlation, however, implies a large degree of redundancy of information in components. In reality, there exist varying degrees of correlation amongst the different components. Therefore the actual  $FRR \in ([R_C]; \sum R_c)$ . The same argument applies to the true rejection rate. Ideally, we want as little correlation as possible amongst components to minimize information redundancy whilst simultaneously maximizing  $|S_1^F \dots \cup \dots S_C^F|$  and minimizing  $|S_1^A \dots \cup \dots S_C^A|$  where  $S_c^F$  is the set of forgeries rejected by the  $c^{th}$  component.

If there is a large number of components, e.g. the 18 features in the single-variate signal memory of Section 4.5.3 involved in a decision, one might allow a small number of them to fail and yet conclude that the signature is authentic. Of course, allowing more components to fail will reduce the FRR but will at the same time increase the FAR. Such a scheme then proves useful only if the reduction in FRR outweighs the increase in FAR. Figure 4.23 shows a histogram of the number of components rejecting signatures in a single verification attempt for a certain experiment. As the plot illustrates, allowing up to 5 rejections out of the 18 components will result in the correct classification of a number of authentic signatures which would otherwise have been rejected. This comes at the price of accepting a relatively small

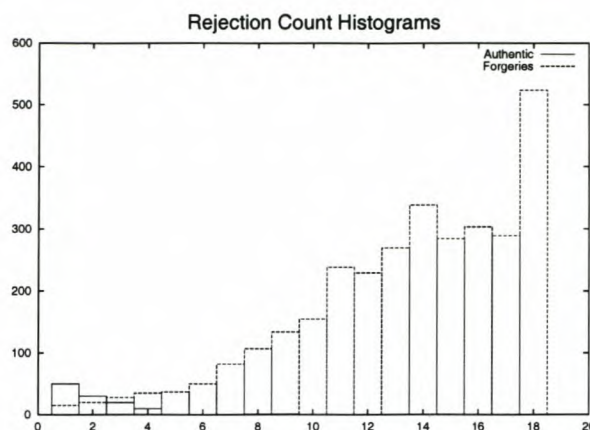


Figure 4.23: **Rejection Count Histogram:** The plot shows the histograms for the number of individual components rejecting authentic and forged signatures.

number of *good* forgeries being rejected by less than five components.

The results attained by the methods described above are presented in Chapter 5 for comparison.

## 4.7 Signature Verification Interface and Database

[51] states two criteria by which to evaluate a signature verification system

- when trying the system in person, it must work and
- when testing the system on large databases, it must exhibit low statistical error rates.

To facilitate hands-on testing of the system, we have created a graphical user interface (GUI) shown in Figure 4.24 which enables users to register with the system. Once registered they can enter several signature samples which are saved in a database. The GUI enables the operator to train a model or verify a signature through separate controls. A modular design facilitates testing of different algorithms with the same GUI through a plug-in interface. The GUI works under the X-windowing [5] system on a workstation running the Linux [2] Operating System. It was implemented with the GTK [1] application framework under C/C++. It makes use of a Wacom Intuos [4] tablet to capture signatures.

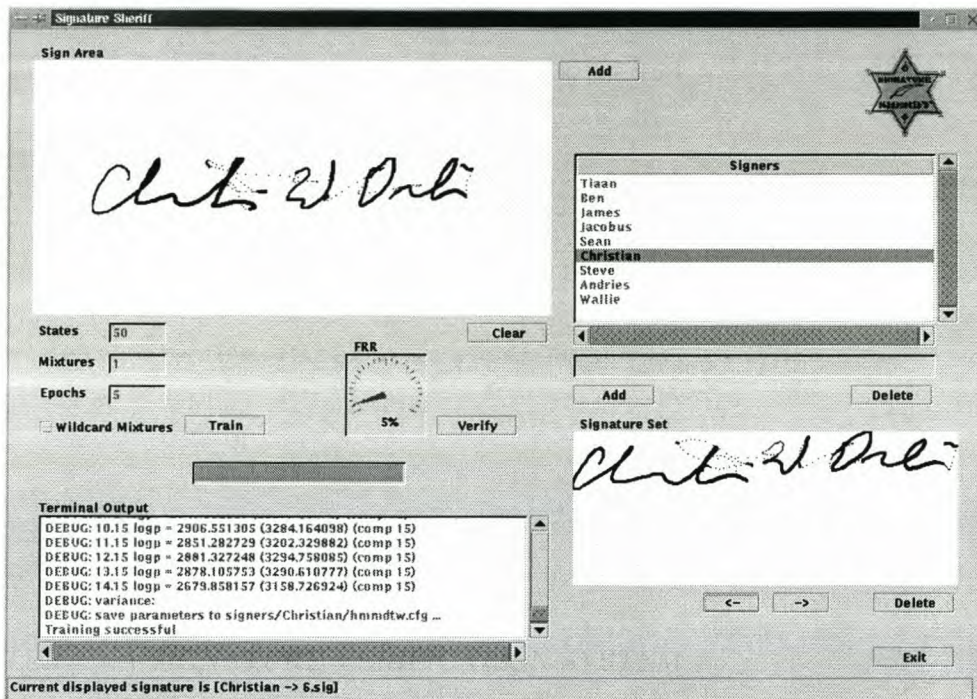


Figure 4.24: **Signature Verification Graphical User Interface:** This application allows the user to enter a training signature set and verify a signature for authenticity.

Tablets are generally used as high precision input devices for Computer Aided Design (CAD) applications (see Section 2.3.1). In these programs, a user looks at a high resolution workstation display while positioning an on-display cursor with the tablet pen. This scenario unfortunately does not correspond with the way we render signatures. We have experienced that writers find it very difficult and unnatural to look at a computer display while signing elsewhere on a tablet. According to [53], signing is a learned process; it is the result of a ballistic motion with essentially no visual feedback. Our experience leads us, however, to believe that this claim underestimates the importance of visual feedback during signing. The participants in our experiments almost unanimously agreed that the lack of visual feedback from the tablet surface makes it more difficult to sign consistently. Rather than focussing on the display, writers focus on the tablet surface and have to imagine the partial signature already created while signing. It is interesting to note that the dynamics of a signature, such as the velocity profile, are not affected as badly by the lack of visual feedback as are the statics, e.g. signature shape. Some tablets possess an LCD display which serves as a secondary display and obviates the need for the user to look elsewhere while signing.

As described in Section 2.2, compiling a research signature database representative of a possible target population is a resource intensive task. The further need for quality forgeries makes this even more so. We therefore obtained permission from J.G.A. Dolfing, the author of amongst others [23], to use a database compiled at his institution for our research for which we are greatly appreciative.

This database contains 1530 authentic signatures for 51 individuals of who 45 are males and 6 females. Each individual donated 30 signatures which are divided into two sets of equal size for training and validation purposes. Each stored signature comprises of a sequence of 5-tuples (1) pen x, (2) pen y, (3) pressure, (4) x tilt and (5) y tilt sampled from the PAID tablet described in Section 2.3.1. Three types of forgeries are included in the database (1) home-improved, (2) over-the-shoulder and (3) professional. The home-improved and over-the-shoulder forgeries were created by the 51 individuals which contributed the authentic signatures. Each data acquisition session involved both a signer and a forger. While the signer contributed signatures, the forger closely watched the dynamics of the signing process. The forger then attempted to recreate the dynamics of the observed signature to create the over-the-shoulder forgery. The roles were then reversed to produce another set of signatures and forgeries. Each individual was provided with a paper copy of another signature to take home for practice. A set of home-improved forgeries were subsequently donated by those individuals. These forgeries can be regarded as amateur and the final home-improved set consists of 1530 forgeries whereas the over-the-shoulder set contains 1470 forgeries. Additionally, four forensic document examiners provided a total of 270 forgeries of 20 individuals from the database. The signatures of these individuals were evenly divided over complexity classes *easy*, *moderately easy* and *difficult* to forge.

## 4.8 Summary

In this chapter, we have described various problems associated with the development of an ASV system and solutions to some of them. In particular, we have presented a solution to the problem of rotational invariance of signatures. We have shown how a number of feature signals can be derived from the sampled signature signals among which the concept of

line thickness is new. We have presented a general method to initialize left-right HMMs from training sequences. Furthermore, we have proposed a number of semantically different HMMs to model the signing process. We have shown how signature likelihood values calculated from signature models are standardized to a common reference frame to facilitate the use of a single threshold value. Finally, we have described an application to experiment with the models and the database used to derive the performance results presented in the next chapter.

## Chapter 5

# Performance Evaluation and Comparison

### 5.1 Overview

This chapter documents the results of our experiments as described in Chapter 4. The software used in the experiments were written by us in C++, the Korn Shell, Perl and AWK Scripting under the Linux Operating System running on a Pentium III 600Mhz computer.

The results are presented in tabulated form. In the good spirit of science, a number of abbreviations are used in order to make these tables as compact as possible. These are

- #K—number of Gaussian kernels used in density functions at HMM states,
- $S_x$ — $x$  combined single-variate HMMs,
- $M_x$ —single  $x$ -dimensional multi-variate HMM,
- $M_x M_y$ — $x$  combined  $y$ -dimensional multi-variate HMMs,
- Z() $\text{—}$ standardization i.e.  $z = \frac{x-\mu}{\sigma}$ ,
- LL—log-likelihood metric,
- NLL—length-normalized log-likelihood metric,



- OLD—distance to optimal log-likelihood metric normalized by length,
- L—likelihood metric (LL/NLL/OLP ?),
- $TD_x$ —threshold decision function with  $x$  allowed failures (SV only),
- PRM—parametrization (time/arc-length ?),
- A—arc-length parametrization of signals,
- T—time parametrization of signals,
- PU—pen-up (is pen-up information included ?),
- N—narrow (transition distance and termination state set are small),
- W—wide (transition distance and termination state set are large),
- RR—reduction ratio compared to original data size,
- TT—training time with 15 exemplars (in seconds),
- VT—verification time of 1 signature (in seconds),
- DTW—dynamic time warping initialization,
- RND—random initialization,
- TRN—authentic training signature,
- TST—authentic testing signature,
- FHI—home-improved forgery,
- FPR—professional forgery and
- FSH—over-the-shoulder forgery.

Table 5.1 provides a summary of the best performance levels attained by the different experiments conducted in this study. The lowest EER of 0.6% is achieved by the unreduced signal memory model (built from arc-length parametrized signatures). However, it comes at the price of substantial training time involved to fix the model parameters on the data.

Performance Summary							
Experiment	RR	TT	VT	EER	FAR for FRR		
					1%	3%	5%
Signal Memory	1	60	1	0.6	0.6	0.6	0.5
Signal Memory	2	20	< 1	0.8	0.8	0.5	0.4
Signal Memory	3	7	< 1	1.0	1.0	0.7	0.7
Statistical	4	3	< 1	4.5	11.5	5.4	4.7
Specialized	10	2	< 1	3.3	8.0	3.9	3.2
Random	1	30	1	3.5	8.5	3.5	2.7

Table 5.1: **Performance Results:** This table presents a summary of the performance levels attained during the verification experiments described in this study.

The specialized descriptor model achieves a moderate EER of 3.3% in an acceptable training time. A good compromise seems to lie in the signal memory model reduced three-fold which achieves an EER of roughly 1% with an average training time of seven seconds per model.

We now weigh the various factors present in decision making against each other to draw conclusions as to the most *affordable* signature verification model in as far as HMMs are concerned.

## 5.2 Threshold Decision Functions

As described in Section 4.6, the threshold decision function refers to the way we interpret the model output to reach a conclusion on the authenticity of a signature. For multi-variate models where a single likelihood metric value is involved, the decision function simply entails comparing the likelihood value to a threshold value and depending on this outcome, we reach a conclusion. For experiments where the output from multiple models are combined to make a decision, we have the option of allowing a small number of failures in individual comparisons and yet conclude the signature to be authentic.

As Table 5.2 demonstrates, allowing only one rejection significantly reduces the EERs of the experiments. The number of rejections which will improve the performance depends on the distribution of rejection counts and the number of components in the combination. For

18 Combined Single-Variate Signal Memory												
PRM	PU	TT	VT	Z(L)	EER					FAR for FRR		
					TD <sub>0</sub>	TD <sub>1</sub>	TD <sub>3</sub>	TD <sub>5</sub>	TD <sub>10</sub>	1%	3%	5%
T	Y	90	1	LL	2.9	1.3	1.3	1.2	1.3	1.3	1.1	0.8
T	Y	90	1	NLL	2.9	1.2	1.2	1.1	1.2	1.2	1.1	0.7
T	Y	90	1	OLD	2.9	1.6	1.7	1.5	1.7	1.7	1.4	0.9
T	N	60	1	LL	1.5	1.5	1.4	1.5	1.5	1.6	1.1	0.6
T	N	60	1	NLL	1.5	1.1	1.1	1.1	1.1	1.1	0.9	0.6
T	N	60	1	OLD	1.5	1.2	1.1	1.1	1.1	1.2	1.0	0.7
A	Y	90	1	LL	2.5	1.2	1.0	1.0	1.0	1.0	0.9	0.8
A	Y	90	1	NLL	2.4	1.1	0.9	0.9	0.9	0.9	0.9	0.8
A	Y	90	1	OLD	2.4	1.1	1.1	1.0	1.0	1.0	0.9	0.7
A	N	60	1	LL	2.0	0.8	0.9	0.9	0.9	0.8	0.5	0.3
A	N	60	1	NLL	2.0	0.6	0.7	0.7	0.7	0.6	0.6	0.5
A	N	60	1	OLD	2.0	0.7	0.8	0.8	0.8	0.7	0.6	0.4

Table 5.2: **Performance Results:** This table presents the error levels achieved by the unreduced single-variate combination signal memory experiments described in Section 4.5.3. The experiments were conducted on both time and arc-length parametrized signatures where pen-up sections were left intact and subsequently removed. The EERs are calculated from the entire set of forgeries, i.e. all types.

the experiments reported in Table 5.10, allowing any rejections significantly deteriorated the performance as there is a small number of components in the combinations.

### 5.3 Likelihood Metrics

Recall from Chapter 3 that HMM theory allows us to calculate a likelihood value for an input sequence given a model. In Section 4.6.2, we have proposed the normalization of this likelihood value through division by the length of the input sequence to minimize the effect it has on the likelihood value as can be seen when inspecting the relevant calculations. We furthermore calculate the distance of the likelihood to what the optimal likelihood value for a sequence of similar length would be.

As can be seen from results presented in Table 5.2, the NLL metric yields the lowest equal error rates. The average number of samples in the signatures encountered in this study are

- Authentic training—TRN—282,
- Authentic testing—TST—281,
- Forged home-improved—FHI—398,
- Forged over shoulder—FSH—331 and
- Forged professional—FPR—468.

(see Section 4.7 for a description of the signature database). From this we can see that the forgeries are generally much longer than authentic signatures. A forgery could thus result in a likelihood value in the range of authentic signatures due to the accumulative effect of its length during the likelihood calculation. The NLL minimizes this artifact with positive effect as can be seen from the reported figures. The idea of the OLD is also an attempt to curb the influence of the sequence length by using an adaptive reference value (in this case the optimal likelihood value for a sequence of the same length). For the experiments tabulated in Table 5.2, the OLD did not quite succeed to convince that it is in fact an improvement over LL, however, when we look at Tables 5.11 to 5.13 it consistently performs better than the LL metric.

## 5.4 Stability of Pen-up Information

Recall that the signature acquisition hardware described in Section 4.2 can measure pen movement even when the pen is not in contact with the surface of the tablet. When looking in Table 5.2 at the results for models with regard to pen-up information (column demarcated by PU), we can see that those excluding these parts of the sampled signals, generally perform better than those including it. This brings us to believe that pen-up information is not stable as a feature and should therefore be excluded from verification models. Figure 4.12 confirms this by showing the pen-up sections being fairly unstable even for a signer with a fairly stable visible signature manifestation.

## 5.5 Parametrization

As Section 4.4 describes, we do not need necessarily have to work with a time parametrization of the signals (as they are sampled). An arc-length parametrized representation can be derived from a time parametrization and this version then used to train a model instead.

From Table 5.2 we can see that the models built from arc-length parametrized signatures consistently outperforms their time-parametrized counterparts. They achieve a lowest EER of 0.6% whereas the time-parametrized models achieved a lowest of 1.1%.

When looking at Table 5.3, we can see how the components consistently perform better under an arc-length parametrization with the exception of centripetal and tangential acceleration. Under a time parametrization these components' discriminative power improved drastically. On the other hand, the path tangent shows a significant improvement under an arc-length parametrization whereas some components are not affected at all.

As there is nothing which prevents us from forming a hybrid combination of models, we have replaced the arc-length parametrized centripetal and tangential acceleration components with that of the time parametrized versions and left the other components in arc-length parametrized form. This combination succeeded in achieving an EER of 0.6% and thus failed to improve on the pure arc-length parametrized combination. This shows us that when combining the output of a number of models, individual component performance cannot be regarded alone but the correlation amongst components need to be inspected as well.

## 5.6 Signal Reduction

When looking at the TT (training time) column of Table 5.2, we can see that the low error rates achieved by the models in these experiments come at a price. It takes relatively long to perform the re-estimation training on these models due to their large number of parameters. Even though these times might be reduced by optimization of the verifier implementation, it is doubtful whether it could be lowered to times feasible for practical deployment where

Parametrization Comparison		
Experiment	Arc-Length	Time
1. X	75	75
2. Y	71	71
3. Pressure	72	71
4. Tilt X	43	40
5. Tilt Y	40	39
6. Longitude	38	36
7. Latitude	44	43
8. $D_{Pressure}$	53	53
9. Velocity X	85	81
10. Velocity Y	83	80
11. Acceleration X	80	77
12. Acceleration Y	78	80
13. Velocity	77	76
14. Path Tangent	90	84
15. $A_{Tangential}$	75	77
16. $A_{Centripetal}$	63	74
17. Acceleration	74	72
18. Line Thickness	85	85

Table 5.3: **Parametrization Comparison:** This table shows the forgery rejection ability for the components under an arc-length and time parametrization.

Reduced Single-Variate Signal Memory							
Reduction	TT	VT	N/W	EER	FAR for FRR		
				$TD_x$	1%	3%	5%
1	60	1	N	0.6	0.6	0.6	0.5
1	180	> 1	W	0.9	0.9	0.7	0.6
2	20	< 1	N	0.8	0.8	0.5	0.4
3	7	< 1	N	1.0	1.0	0.7	0.7

Table 5.4: **Performance Results:** This table presents the results for HMMs trained on data reduced by resampling on sparser intervals. It also shows the effect of narrow and wide parameter settings for the models.

potentially thousands of users will have to be supported.<sup>1</sup>

Table 5.4 shows the results for signal memory models built from sparsely sampled signatures. For a three-fold reduction, the EER increases to 1.0%, which is still an acceptable figure. At the same time we can see how the required training time has been reduced significantly and verification time becomes less than a second.

This table highlights another aspect which needs to be taken into consideration when configuring a HMM. The more states can be reached from a given state and the larger the set of terminal states, the more parameters are used to define the model and the more memory and CPU time are needed for storage and processing. The question arises whether more parameters necessarily lead to better discrimination. From the results presented in the table, we can see how the training time increased drastically and the performance level deteriorated. The lower performance level can be attributed to the higher amount of variance built into the model. This means that the model will now accept more authentic signatures but also accept more forgeries as the margin of digression is increased by the widening of the model parameters. An analogy can be drawn to the field of artificial neural networks where an increase in the number of neurons often result in over-generalization.

<sup>1</sup>What constitutes a *feasible time* is a somewhat difficult question to answer without deployment scenario information. Thus, the time information presented here should rather be viewed from a comparative perspective.

State Duration Restrictions			
			EER
Restricted	TT	VT	TD <sub>x</sub>
N	50	1	2.7
Y	60	1	0.6

Table 5.5: **Performance Results:** This table presents the results for HMMs with state duration restrictions disabled and enabled.

## 5.7 State Duration Restrictions

Table 5.5 shows that the simple state duration restrictions proposed in Chapter 3, result in a large performance improvement. This can be attributed to the larger amount of control we have over what a matching input sequence might look like. As previously shown, the number of samples in forgeries are generally more than in authentic signatures. These differences are partially handled by the velocity component model; however, the excessive number of samples has to be absorbed somewhere in the model. Without state duration restrictions, these samples are absorbed through state self-transitions. With duration modelling, we can now restrict the time spent in a state to what was observed from training sequences and no more. We can furthermore see that enabling duration restrictions results in an increase in training time which is, however, not proportional to the increase in performance.

## 5.8 Combination of Discriminative Components

In an attempt to reduce model sizes and processing times without a significant penalty in performance, we investigated the discriminative ability of individual components in an attempt to combine only a subset with good discriminative abilities.

Table 5.6 shows the percentage of detected forgeries by each individual component as well as the percentage of rejected authentic signatures. The percentages were taken from the signal memory experiment which yielded an EER of 0.6%. One should note that the numbers refer to percentages based on rejected counts only. Therefore, x% authentic rejection does not mean that x% of all authentic signatures were rejected by the component but rather that of



Component Analysis				
Experiment	% Forgeries Detected		% Authentic Rejected	
	Overall	Solely	Overall	Solely
1. X	75	0	15	3
2. Y	71	0	12	0
3. Pressure	72	0	3	0
4. Tilt X	43	0	3	0
5. Tilt Y	40	0	3	0
6. Longitude	38	0	0	0
7. Latitude	44	0	3	0
8. $D_{Pressure}$	53	0	3	0
9. Velocity X	85	0	21	6
10. Velocity Y	83	0	9	0
11. Acceleration X	80	0	21	0
12. Acceleration Y	78	0	18	0
13. Velocity	77	0	21	9
14. Path Tangent	90	0	67	45
15. $A_{Tangential}$	75	0	15	0
16. $A_{Centripetal}$	63	0	9	0
17. Acceleration	74	0	18	3
18. Line Thickness	85	0	3	0

Table 5.6: **Component Discriminative Power:** This table presents the discriminative power of the individual components as described in Section 4.5.3 seen from different viewpoints. Percentages were rounded to the nearest percentage point and taken at the point of EER equal to 0.6% for the model deduced from arc-length parametrized data with pen-up sections removed.

the small percentage of authentic signatures rejected, the component's contribution made out  $x\%$ .

From the raw sampled components, we see that visual information and pressure possess fairly strong discriminative ability. Our findings however seem to contradict the emphasis placed by [23] on pen tilt information. The path tangent manifested as the most discriminative component. Line thickness performed well in the sense that it rejects a large amount of forgeries yet rejects very few authentic signatures. None of the components detected a significant amount of forgeries on their own. The path tangent solemnly made a significant contribution to the number of rejected authentic signatures. This number is still a very small absolute number compared with the large number of forgeries rejected.

Apart from inspecting the individual discriminative powers of components, we need to know how much overlap exists in the abilities of components. One would gain little by combining two components with high discriminative ability but which detect exactly the same set of forgeries. Such a combination will only increase the model size without improving on the FAR.

Table 5.7 shows the amount of overlap between the individual components in matrix form. An entry (i,j) should be read that component i and component j concurrently detect  $x_{ij}$ % of the forgeries. So for instance, components 9 and 14 concurrently detect 79% of the forgeries. To know what the combined performance would be, we need to calculate the difference between this percentage overlap and the percentage of detected forgeries by each constituent component. When these two differences are added to the overlap percentage, we arrive at the combined performance level for the two components. We can see how a combination of components 9 and 14 would yield a forgery detection rate of  $79+11+6=96\%$  i.e. a 4% FAR. Visual information only, i.e. x and y, yield a combined FAR of 14%. True velocity i.e. speed (13) and direction (14), yield a combined FAR of 6% when the FRR is less than 1%.

Tables 5.8 and 5.9 tabulate the differences and combined performances. We use these percentages to guide us in combining two components but also to guide us as to which combinations of more than two components could yield a higher performance level without resorting to similar but more complex bookkeeping techniques. Component 14 (path tangent) plays a pivotal role in the combination models as can be seen from its column in Table 5.9. Table 5.10 presents results for a possible combination we have found to perform well. The combination contains 4 of the raw sampled components which illustrates that they show little correlation as can be expected. The velocity in the x-direction and the path tangent are combined with the raw sampled components. We can see from the table that an EER of 1.9% can be attained with an average model training time of 18 seconds with unreduced data. For this combination, data reduced by a third attains an EER of 3.0%. Looking at these results and those from Table 5.6, we would thus rather use a combination of all the components from a sparsely sampled dataset to decrease model training times than use a small number of discriminative components from an unreduced dataset in an attempt to cut on training times.

Overlap Matrix																		
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18
01	75	59	58	39	35	34	40	45	68	67	66	62	62	70	61	54	61	68
02	59	71	55	37	34	33	37	41	63	65	62	59	59	67	58	51	57	65
03	58	55	72	36	35	33	37	49	66	65	64	61	62	68	61	50	60	68
04	39	37	36	43	26	30	32	29	40	40	39	36	38	41	37	33	37	40
05	35	34	35	26	40	26	31	29	38	38	37	35	36	38	35	31	35	38
06	34	33	33	30	26	38	27	26	36	36	35	34	34	37	33	30	34	36
07	40	37	37	32	31	27	44	31	41	41	40	37	40	42	39	35	38	42
08	45	41	49	29	29	26	31	53	51	51	50	47	49	51	48	39	48	51
09	68	63	66	40	38	36	41	51	85	77	76	72	73	79	72	58	69	77
10	67	65	65	40	38	36	41	51	77	83	75	74	74	80	71	59	70	77
11	66	62	64	39	37	35	40	50	76	75	80	71	70	76	70	58	68	75
12	62	59	61	36	35	34	37	47	72	74	71	77	68	74	68	55	66	71
13	62	59	62	38	36	34	40	49	73	74	70	68	77	73	68	54	67	72
14	70	67	68	41	38	37	42	51	79	80	76	74	73	90	71	61	69	80
15	61	58	61	37	35	33	39	48	72	71	70	68	68	71	76	55	67	69
16	54	51	50	33	31	30	35	39	58	59	58	55	54	61	55	63	53	59
17	61	57	60	37	35	34	38	48	69	70	68	66	67	69	67	53	74	68
18	68	65	68	40	38	36	42	51	77	77	75	71	72	80	69	59	68	85

Table 5.7: **Component Correlation Matrix:** This table presents the correlation amongst the individual components as far as concurrently rejected forgeries are concerned. The percentages are forgery detection rates for FRR less than 1%.

Overlap Difference																		
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18
01	0	15	16	36	40	41	35	30	7	8	9	13	13	4	13	21	14	7
02	11	0	15	33	36	37	33	29	7	6	9	11	12	3	13	19	13	6
03	13	16	0	36	37	39	35	22	6	6	7	11	10	4	11	22	11	3
04	4	5	7	0	16	13	11	14	2	3	4	6	5	1	6	9	6	2
05	5	5	5	13	0	14	9	11	2	2	3	5	4	1	5	9	5	2
06	4	5	5	9	12	0	11	12	3	2	3	5	4	1	5	8	4	2
07	5	7	8	12	14	17	0	14	3	3	4	7	5	2	5	9	6	2
08	8	12	4	24	25	27	22	0	2	3	3	6	4	3	6	14	5	2
09	17	22	19	45	48	50	44	34	0	9	9	14	12	6	14	28	16	8
10	16	18	18	44	45	47	42	33	7	0	8	10	10	4	12	24	13	6
11	15	19	16	42	43	45	40	31	5	6	0	10	10	4	10	23	12	6
12	15	18	17	41	43	44	40	31	6	4	6	0	9	4	10	23	12	6
13	15	18	15	39	41	43	37	28	4	3	7	9	0	4	9	23	10	5
14	20	23	22	49	52	53	48	40	11	11	14	17	17	0	19	29	21	10
15	14	18	15	39	40	42	37	28	4	5	5	8	7	5	0	21	8	6
16	9	12	13	30	32	33	28	24	5	4	5	9	9	2	9	0	10	5
17	13	17	13	37	39	40	36	26	5	4	6	8	7	5	7	20	0	6
18	17	20	16	44	46	49	43	34	7	8	10	13	13	4	15	26	17	0

Table 5.8: **Component Overlap Difference Matrix:** This table presents the difference between the correlation percentage and the percentage detected by the component heading the column.

Combined Performance																		
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18
01	75	86	88	79	79	79	79	83	92	91	90	90	90	95	89	84	88	92
02	86	71	87	76	76	75	77	82	93	89	89	89	89	94	88	82	87	91
03	88	87	72	78	77	77	79	75	91	90	88	88	87	94	87	85	85	88
04	79	76	78	43	56	51	55	67	88	86	84	84	82	91	81	73	80	87
05	79	76	77	56	40	52	53	64	87	85	83	82	81	92	80	72	79	86
06	79	75	77	51	52	38	55	65	88	85	83	82	81	91	80	71	78	87
07	79	77	79	55	53	55	44	67	89	86	85	84	82	92	81	72	80	87
08	83	82	75	67	64	65	67	53	87	86	84	84	81	93	81	77	79	87
09	92	93	91	88	87	88	89	87	85	92	90	91	89	96	89	91	90	93
10	91	89	90	86	85	85	86	86	92	83	89	87	87	94	88	87	87	91
11	90	89	88	84	83	83	85	84	90	89	80	87	87	95	86	86	86	90
12	90	89	88	84	82	82	84	84	91	87	87	77	86	94	85	86	85	91
13	90	89	87	82	81	81	82	81	89	87	87	86	77	94	84	86	84	90
14	95	94	94	91	92	91	92	93	96	94	95	94	94	90	95	93	95	94
15	89	88	87	81	80	80	81	81	89	88	86	85	84	95	76	84	82	91
16	84	82	85	73	72	71	72	77	91	87	86	86	86	93	84	63	83	89
17	88	87	85	80	79	78	80	79	90	87	86	85	84	95	82	83	74	90
18	92	91	88	87	86	87	87	87	93	91	90	91	90	94	91	89	90	85

Table 5.9: **Combined Component Performance Matrix:** This table presents the percentage of forgeries which will be detected by combinations of two components. The percentages are forgery detection rates for the FRR less than 1%.

<i>x</i> Combined Single-Variate Signal Memory							
				EER		FAR for FRR	
Components	RR	TT	VT	TD <sub>0</sub>	1%	3%	5%
1,2,3	1	9	< 1	3.9	6.0	4.3	3.4
1,2,3,5	1	12	< 1	3.5	5.5	3.8	3.0
1,2,3,5,9	1	15	< 1	2.3	3.5	2.3	1.7
1,2,3,5,9,14	1	18	< 1	1.9	2.3	1.6	1.3
1,2,3,5,9,14	2	6	< 1	2.3	2.8	2.1	1.7
1,2,3,5,9,14	3	3	< 1	3.0	3.8	3.0	2.5

Table 5.10: **Performance Results:** This table presents the results for combinations of different components with strong discriminative characteristics as taken from table 5.9.

## 5.9 Multi-Variate Models

Table 5.11 presents the results for the multi-variate signal memory experiments described in Section 4.5.4. These results can be directly compared to those from Table 5.2. As explained in Chapter 4, the differences between them are that this experiment uses a single model per signer and therefore a single timing function to align all components within the model whereas those from Table 5.2 combines the output from multiple models each with its own separate timing function. To be able to use a single timing function, this experiment employs multi-variate density functions in the models in contrast to the single-variate density functions used in the other experiment.

The fact that a single timing function is used, reduces the size of the models compared to the sum of the equivalent combination of single-variate models and therefore it takes less time to train. The performance levels attained, however, are not as good as that of the combined single-variate models (the column labeled  $S_{18}$  is included for easy comparison). The reason for this can be attributed to the lack of expressive power of a single timing function to align multiple signals which are not aligned by a uniform timing function.

When calculating the likelihood of an  $N$ -dimensional variable, we obtain a value which is an indication of how likely the simultaneous occurrence of a specific combination of  $N$  values are. If a single timing function aligns all the components of a signature, then such a density function will be suitable for modelling the distribution of components at specific points in a signature as the values can be expected to assume a multinomial distribution with little variance, at least for consistent signers. As stated in Section 4.5.4, our experiments with the DTW initialization scheme lead us to believe that the timing function of a single component does not necessarily align the other individual components well (although this is true for the most consistent signers). Figure 4.11 illustrates this argument. With a single alignment function, the variance in the multi-nomial distributions at states becomes exceedingly large. For a distribution with large variance, most component tuples (authentic or not) are reported to be relatively unlikely to occur. As far as the horizontal control is concerned, the model now has difficulty capturing state transitions accurately. The density functions need to assist in finding paths with high likelihood during re-estimation training but as the large variance

results in these functions being relatively flat, the re-estimation training suffers to find a set of highly probable transitions between states which would make it more difficult for forgeries to generate likelihood values close to those of authentic signatures. In other words, the distribution of likelihoods for authentic and forgery signatures against a trained model becomes less disparate as the model outputs are more vague and thus less discernible.

One possible solution to this is to create mixtures of Gaussian kernels to increase the modelling power of the density functions. As Table 5.11 shows, this does not result in a consistent performance improvement and, generally speaking, performance degrades as the number of mixture components increase.

We can now understand the advantage combined single-variate models have over single multi-variate models. The density function at each state provides the likelihood for the component to assume a certain value in that state regardless of what the values of the other components are. The transition probabilities for each single-variate model can be configured separately for the component it models. The more focused density functions can now assist better with the re-estimation of the transition probabilities thereby converging to a more accurate model of the actual signing process. Prior to combining the outputs of the single-variate models during decision making, each output can be standardized separately by a different set of average and deviation values. Finally, we can allow for a small set of component failures and yet conclude that a signature is authentic whereas each component's contribution is embedded in a single likelihood value in the multi-variate model case. All these factors contribute to the lower EER achieved by the single-variate experiments in comparison with equivalent multi-variate experiments.

## 5.10 Signature Segmentation Models

Table 5.12 and 5.13 show the results for the experiments based on segment representations of signatures. These models are independent of any specific parametrization of the signature data as alignment is rather dependent on the segmentation of the signatures at similar places along the trajectory as explained in Section 4.5.6. As can be seen from Table 5.12, the generic statistical segment descriptors achieve less than acceptable error rates with a lowest EER of

18-D Multi-Variate Signal Memory											
PRM	PU	TT	VT	Z(L)	EER (TD)				FAR for FRR		
					(1K/DTW)	(1K/RND)	S <sub>18</sub>	(2K/DTW)	1%	3%	5%
T	Y	20	1	LL	4.7	7.8	1.2	4.4	15.0	7.1	4.2
T	Y	20	1	NLL	2.4	3.4	1.1	3.2	6.0	1.8	0.7
T	Y	20	1	OLD	2.5	3.7	1.5		6.5	2.0	0.7
T	N	15	1	LL	5.8	8.8	1.4	4.0	22.0	13.9	6.7
T	N	15	1	NLL	3.6	5.1	1.1	2.6	15.0	5.7	2.4
T	N	15	1	OLD	3.5	5.1	1.1		15.0	5.9	2.7
A	Y	20	1	LL	3.5	6.9	0.4	4.1	8.0	4.2	2.0
A	Y	20	1	NLL	2.6	5.0	0.4	3.7	7.8	2.2	1.3
A	Y	20	1	OLD	2.8	5.4	0.5		7.9	2.0	1.3
A	N	15	1	LL	4.2	6.9	0.1	4.2	20.0	5.1	3.6
A	N	15	1	NLL	2.9	3.5	0.1	3.3	12.0	2.0	1.2
A	N	15	1	OLD	2.7	3.8	0.1		7.9	2.0	1.4

Table 5.11: **Performance Results:** This table presents the error levels achieved by the multi-variate signal memory experiments described in Section 4.5.4. For comparison, the result of the equivalent randomly initialized models are provided as well as described in Section 4.5.5. The experiments were conducted on both time and arc-length parametrized signatures where pen-up sections were left intact and subsequently removed. The EERs are calculated from the entire set of forgeries, i.e. all types.



18 Combined 4-D Multi-Variate Segment Statistics								
				EER		FAR for FRR		
PU	TT	VT	Z(L)	TD <sub>5</sub> (1K)	TD <sub>5</sub> (2K)	1%	3%	5%
Y	3	< 1	LL	6.1	6.0	17.6	8.1	7.2
Y	3	< 1	NLL	4.5	4.6	11.5	5.4	4.7
Y	3	< 1	OLD	4.6	4.6	11.7	5.3	4.5
N	2	< 1	LL	6.2	6.2	19.0	9.5	8.0
N	2	< 1	NLL	5.3	5.4	12.7	8.3	5.7
N	2	< 1	OLD	5.4	5.5	15.0	9.9	6.4

Table 5.12: **Performance Results:** This table presents the error levels achieved by the statistical segment descriptor experiments described in Section 4.5.6. The experiments were conducted on signatures where pen-up sections were left intact and subsequently removed. The descriptors in this experiment are independent of the parametrization. The EERs are calculated from the entire set of forgeries, i.e. all types.

4.5% and is as such not of much use for signature verification. The results for the specialized segment descriptors presented in Table 5.13 are more promising with an EER of 3.3%. The statistical descriptors reduce the signature data on average by 75% whereas the specialized descriptors reduce it by 90%. Yet, the latter achieve a lower EER. This stresses the general notion that to achieve results better than generic solutions do, domain specific knowledge has to be employed. As is the case with the other experiments, the models excluding pen-up information performed better than those including it reinforcing the notion that the instability of this information does harm to verification performance. Again, increasing the number of mixtures in the density functions does not result in any significant performance improvement.

## 5.11 Summary

This chapter reported on the results of our experiments and provided explanations for some of the observed behaviour. When looking at Table 5.11 we can see that the results for randomly initialized models are consistently worse than that of equivalently configured experiments initialized by values taken from aligned training sequences. This supports the widely held belief that HMMs need to be initialized with sensible initial values in order for re-estimation training to converge to a good solution.

22 Combined Single-Variate Specialized Descriptors								
					EER	FAR for FRR		
PRM	PU	TT	VT	Z(L)	TD <sub>7</sub>	1%	3%	5%
T	Y	2	< 1	LL	5.3	10.8	6.1	5.7
T	Y	2	< 1	NLL	4.7	9.0	5.3	4.5
T	Y	2	< 1	OLD	4.1	10.8	7.2	4.0
T	N	2	< 1	LL	4.4	8.0	5.0	4.2
T	N	2	< 1	NLL	3.7	7.9	4.1	3.3
T	N	2	< 1	OLD	3.3	8.0	3.9	3.2

Table 5.13: **Performance Results:** This table presents the error levels achieved by the specialized segment descriptor experiments described in Section 4.5.7. The experiments were conducted on time parametrized signatures only where pen-up sections were left intact and subsequently removed. The descriptors in this experiment are independent of the parametrization. The EERs are calculated from the entire set of forgeries, i.e. all types.

## Chapter 6

# Conclusions and Directions for Future Research

### 6.1 Conclusion

The main focus of this thesis has been to investigate the applicability of HMMs to the field of ASV. HMMs have been applied to this domain in [71, 23, 36, 48] with promising results. As HMMs are a generic modelling tool, they allow for much variation in the way they are applied; thus, they lend themselves to a number of possible extensions. We have explored a number of different aspects of the modelling process with regard to handwritten signatures.

In Chapter 5, we have presented the results of our experiments as described in Chapter 4. These results have enabled us to draw a number of conclusions of relevance to ASV and in particular when performed by HMMs.

We have shown that zero pressure information (i.e. pen-up sections) is deemed to be unstable and has a negative impact on verifier performance when included in training data. We should point out though that we believe it might very well serve a discriminating role in other approaches to ASV when considering the order of events with regard to *dotting and crossing t's*. This is however performed implicitly by our models as the individual pen-down segments need to be presented in a chronological fashion in order for high-likelihood matches with the

HMMs to be possible.

The arc-length parametrization of signatures proved to be more stable than their time-parametrized counterparts and resulted in higher verifier performance when HMMs were built from such signatures. This could be partially attributed to the high concentration of samples present in areas of low velocity with a time-parametrization, being sampled sparser by the arc-length parametrization which provides a more uniform spread of samples along the pen trajectory.

We have distinguished between single multi-variate models and combinations of multiple single-variate models. The comparison of these results leads us to believe that the increased freedom provided by combining the results of multiple models, plays an important role in increasing verifier performance. Individual component likelihoods can be standardized separately and the timing information of the different components can each be captured by their own HMM transition variables.

Combining output from multiple models enables us to allow for a small number of rejections amongst the individual models, and still conclude a signature to be authentic. This leads to an increase in verifier performance for most of the combined models.

We have presented a number of feature signals and tabulated their individual discriminative performance. Here, we observed that the path tangent possesses the highest degree of discriminative power when modelled with the signal memory HMMs. We have proposed the concept of line thickness which distinguishes well between authentic and forged signatures. The tracking of combined component performances assisted us in forming combinations of models with fewer components whilst still performing fairly well.

In an attempt to reduce the size of models, we have investigated both signal memory models built from sparsely sampled signatures and sequences of segmentation descriptors. It was shown that we can achieve a significant reduction in model size and yet achieve acceptable performance levels by these methods.

We have also shown that a performance increase can be achieved by controlling the number of self-transitions in states and restricting the set of terminal states when performing Viterbi alignments. These two actions make the HMMs more predictable which is an important

consideration in the field of ASV.

From our results, we saw that increasing the number of kernels in the density functions at states, and allowing more freedom in transitions, does not improve performance but generally leads to a deterioration thereof.

A general HMM initialization scheme based on DTW was presented and comparison of results with randomly initialized models confirmed the success of this approach.

Apart from the modelling of signatures, we have also proposed a method for making signatures rotationally invariant by maximizing the variance in the direction of one of the principal axes.

A verification front end was developed using a WACOM digitizer tablet. We have furthermore benchmarked our work against a database used in [23] and achieved an EER of 1% which compares well with the results achieved in that study.

## 6.2 Directions for Future Research

During this study, we have noted a couple of ideas relevant to our work which might warrant further investigation. These can be described as follows.

### 6.2.1 Weighted Decision Function

The size of the cursed area in Figure 4.18 differs for each component. This is due to their varying discriminating ability. Based on this evidence, we can establish a weighted decision function where the size of the cursed area is an inverse indication of our confidence in its ability to decide on signature authenticity. As in the other decision functions, we compare log-likelihood values to a threshold  $\tau_1$ . If a value is smaller than the threshold we add its weight to a forgery belief value. This value is subsequently compared to a fixed threshold value  $\tau_2$  e.g. 1 implying that the weights are chosen in such a way that an accumulated value of 1+ signifies a forgery. This can be expressed as

$$\sum_{c \text{ rejected}}^C w_c <> \tau_2$$

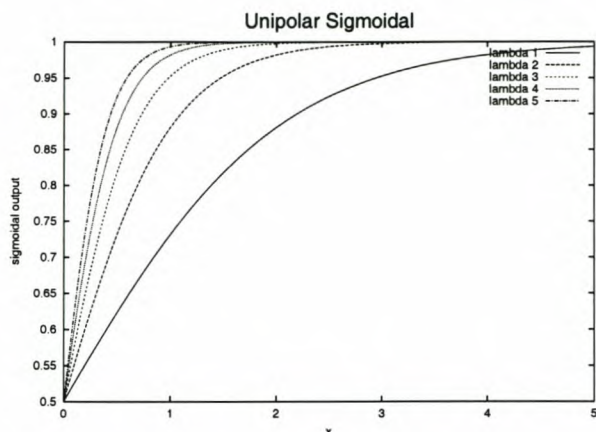


Figure 6.1: **Sigmoidal Function:** The plot shows how the steepness of the sigmoidal is regulated by the value of  $\lambda$ .

Moreover, the decision function can be extended to be forgiving to log-likelihoods very close to the threshold  $\tau_1$  by centering a sigmoidal function at this threshold value. The component weight is then multiplied by the sigmoidal of the log-likelihood's distance from the  $\tau_1$ . Thus, the decision function above becomes

$$\sum_{c \text{ rejected}}^C w_c S(\tau_1 - Z(\log(l_c)), \lambda) <> \tau_2$$

with  $S$  being the unipolar sigmoidal or soft limiter

$$S(x, \lambda) = \frac{1}{1 + e^{-\lambda x}}$$

As Figure 6.1 illustrates, the value of  $\lambda$  dictates the steepness of the sigmoidal function as it rises to 1.

We have attempted to use such a decision function to lower the attained EERs, but failed to obtain convincing evidence. This is partly due to the lack of guidance in finding suitable values for the thresholds  $\tau_1$  and  $\tau_2$ ,  $\lambda$  in such a decision function. It might prove fruitful to find an algorithmic approach to deduce a set of optimal values for these parameters to improve on the simple threshold decision function.

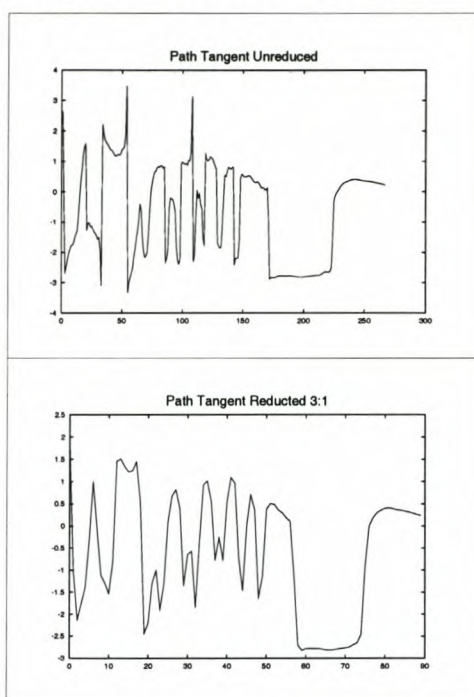


Figure 6.2: **Path Tangent Reduction:** These plots show the path tangent signal deterioration under reduction.

## 6.2.2 Adaptive Sparse Resampling

As Figure 6.2 shows, the path tangent signal quality deteriorates rapidly under reduction. This can be attributed to the steep slopes inherent in signals of this component.

One would have hoped for a resampling approach which maintains the maxima and minima of the signal as much as possible. The filter approach we use dampens the signal at these points to a certain extent. The number and offsets of these points differ amongst the different components of a signature, thereby making it exceedingly hard to maintain the original synchronization of the components during resampling. This becomes a problem only when multi-variate models are built from the signals. For combinations of single-variate models, we need not have models with the same number of states as long as we normalize each individual model's output. Such adaptive sparsely resampled signatures could lead to an improvement in the models initialized from them.

### 6.2.3 Piecewise Continuous HMM

The left-right HMMs used to model signals in this study, memorize the value of signals at discrete points along their length. We could attempt to build a piecewise continuous HMM by performing linear interpolation of the average and deviation between successive states. We can now try to find a value for  $t$  in

$$e^{-\frac{1}{2}\left(\frac{x-(\mu_j-\mu_i)t}{(\sigma_j-\sigma_i)t}\right)^2} \frac{1}{\sqrt{2\pi}(\sigma_j-\sigma_i)t} \quad 0 \leq t \leq 1$$

which maximizes the density function with average and deviation interpolated from the values of state  $i$  and  $j$ . Depending on whether  $t$  is larger or smaller than 0.5, one can classify it in state  $i$  or  $j$  so as to have a transition probability to use in the Viterbi calculations. Alternatively, one could use linear interpolation to obtain transition probabilities once  $t$  is calculated. In regions of constant gradient in the training signals, one can remove a number of samples in this region and use the values at the start and end of such regions to perform the linear interpolation. This stems from the observation that signals often show the most variance around inflection points and not in the regions rising and falling to such points. One could achieve a significant reduction in model size in this way, and it would be interesting to see what the impact on performance would be.

We would like to conclude by saying that we have found HMMs to be most suitable to the task of ASV. This is especially true for the simplest of models which merely act as a memory of the signature signals created possibly from sparsely sampled exemplars. What makes models with such semantics attractive, is their predictable nature. This approach might very well form the core of a production verification system supported by secondary verification checks such as is described in Chapter 2. We believe that ongoing research should focus on finding ways of reducing model sizes without compromising performance and the predictable behaviour of the models.



# Bibliography

- [1] "Gimp Tool Kit website," *www.gtk.org*.
- [2] "Linux operating system website," *www.linux.org*.
- [3] "SMARTPEN biometric authentication system website," *www.smartpen.net*.
- [4] "WACOM graphical tablets website," *www.wacom.com*.
- [5] "X-windows system website," *www.x.org*.
- [6] R. Baron and R. Plamondon, "Acceleration measurement with an instrumented pen for signature verification and handwriting analysis," *IEEE Transactions on Instrumentation and Measurement*, vol. 38, no. 6, pp. 1132–8, 1989.
- [7] L. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, vol. 3, pp. 1–8, 1972.
- [8] L. Baum and J. Egon, "An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology," *Bull. Amer. Meteorol. Soc.*, vol. 73, pp. 360–63, 1967.
- [9] L. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Stat.*, vol. 37, pp. 1554–63, 1966.
- [10] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, vol. 41, no. 1, pp. 164–71, 1970.

- [11] L. Baum and G. Sell, "Growth functions for transformations on manifolds," *Pas. J. Math.*, vol. 27, no. 2, pp. 211-27, 1968.
- [12] Y. Bengio and P. Frasconi, "Input/output HMMs for sequence processing," *IEEE Transactions on Neural Networks*, vol. 7, no. 5, pp. 1231-49, 1996.
- [13] H. Bourlard and N. Morgan, *Hybrid HMM/ANN Systems for Speech Recognition: Overview and New Research Directions*. Springer Verlag, 1997.
- [14] M. Brand, N. Oliver, and A. Pentland, "Coupled Hidden Markov Models for complex action recognition," *MIT Media Lab Perceptual Computing/Learning and Common Sense Technical Report 407*, 1996.
- [15] J. Brault and R. Plamondon, "A complexity measure of handwritten curves: Modeling of dynamic signature forgery," *IEEE Transactions on Systems, Man and Cybernetics*, pp. 400-13, 1993.
- [16] J. Brault and R. Plamondon, "How to detect problematic signers for automatic signature verification," *Proceedings. 1989 International Carnahan Conference on Security Technology*, pp. 127-32, 1989.
- [17] J. S. Bridle, "Alpha-nets: A recurrent 'neural' network architecture with a Hidden Markov Model interpretation," *Speech Communication*, no. 9, 1990.
- [18] J. Bromley, J. Bentz, L. Bottou, I. G. Y. Lecun, C. Moore, E. Sackinger, and R. Shah, "Signature verification using a Siamese time-delay neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 4, pp. 669-88, 1993.
- [19] H. Crane and J. Ostrem, "Automatic signature verification using a three-axis force sensitive pen," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 13, no. 3, pp. 749-70, 1994.
- [20] H. Crane, D. Wolf, and J. Ostrem, "The SRI pen system for automatic signature verification," *Proceedings of the NBS Trends and Applications*, 1977.
- [21] L. de Figueredo, "Adaptive sampling of parametric curves," *Graphics Gems V*, vol. 5, pp. 173-8, 1995.

- [22] G. Dimauro, S. Impedovo, and G. Pirlo, "Component-oriented algorithms for signature verification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 771–93, 1994.
- [23] J. Dolfing, E. Aarts, and J. Oosterhout, "On-line signature verification with Hidden Markov Models," *Proceedings. Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 1309–12, 1998.
- [24] J. du Preez, "Efficient high-order Hidden Markov Modelling," *PhD thesis, University of Stellenbosch*, 1998.
- [25] E. Dudewicz and S. Mishra, "Modern mathematical statistics," pp. 347–67, 1988.
- [26] E. Eernisse, C. Land, and J. Snelling, "Piezoelectric sensor pen for dynamic signature verification," *Proceedings of the IEEE International Electronic Devices Meeting*, 1977.
- [27] M. Fairhurst and P. Brittan, "An evaluation of parallel strategies for feature vector construction in automatic signature verification systems," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 661–78, 1994.
- [28] H. Franco, M. Cohen, N. Morgan, D. Rumelhart, and V. Abrash, "Context-dependent connectionist probability estimation in a hybrid Hidden Markov Model—Neural Net speech recognition system," *Computer Speech and Language*, vol. 8, no. 3, 1994.
- [29] J. Gil and D. Keren, "New approaches to the arc length parameterization problem," in *13th Spring Conference on Computer Graphics* (W. Straßer, ed.), pp. 27–34, 1997.
- [30] J. Gupta and A. McCabe, "A review of dynamic handwritten signature verification," 1997.
- [31] W. Haberman and A. Fejfar, "Automatic identification of personnel through speaker and signature verification—system description and testing," *Proceedings of the Carnahan Conference on Crime Countermeasures*, pp. 20–30, 1976.
- [32] T. Hastie and E. Kishon, "A model for signature verification," *Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics*, vol. 1, pp. 191–6, 1991.

- [33] B. Herbst and D. Richards, "On an automated signature verification system," *IEEE International Symposium on Industrial Electronics*, vol. 2, pp. 600-4, 1998.
- [34] N. Herbst and C. Liu, "Automatic signature verification based on accelerometry," *IBM Journal Research and Development*, vol. 21, no. 3, pp. 245-53, 1977.
- [35] F. T. Johansen and M. H. Johnsen, "Non-linear input transformations for discriminative HMMs," *ICASSP*, 1994.
- [36] R. Kashi, J. Hu, W. Nelson, and W. Turin, "On-line handwritten signature verification using Hidden Markov Model features," *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, vol. 1, pp. 253-7, 1997.
- [37] R. Kashi, W. Turin, and W. Nelson, "On-line handwritten signature verification using stroke direction coding," *Optical Engineering*, vol. 35, no. 9, pp. 2526-33, 1996.
- [38] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464-80, 1990.
- [39] A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler, "Hidden Markov Models in computational biology, applications to protein modeling," *Journal of Molecular Biology*, vol. 235(5), pp. 1501-31, 1994.
- [40] F. LeClerc and R. Plamondon, "Automatic signature verification: The state of the art 1989-1993," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 643-60, 1994.
- [41] L. Lee, "Neural approaches for human signature verification," *ICSP '96. 1996 3rd International Conference on Signal Processing Proceedings*, vol. 2, pp. 1346-9, 1996.
- [42] L. Lee, T. Berger, and E. Aviczer, "Reliable on-line human signature verification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 643-7, 1996.
- [43] R. Lippman, "An introduction to neural networks," *Optical Engineering*, vol. 35, no. 9, pp. 2526-33, 1996.

- [44] S. Lucas and R. Dampier, "Signature verification with a syntactic neural net," *IJCNN International Joint Conference on Neural Networks*, vol. 1, pp. 373–8, 1990.
- [45] T. Matsuura and S. Okamura, "On FIR filter for signature verification," *38th Midwest Symposium on Circuits and Systems*, vol. 1, pp. 366–9, 1996.
- [46] T. Matsuura and H. Sakai, "On stochastic representation of handwriting process and its application to signature verification," *ICSP '96. 1996 3rd International Conference on Signal Processing Proceedings*, vol. 2, pp. 1330–3, 1996.
- [47] T. Matsuura and H. Togiishi, "Two dimensional AR model of signing process and its application to on-line signature verification," *1998 IEEE International Conference on Electronics, Circuits and Systems. Surfing the Waves of Science and Technology*, vol. 2, pp. 545–8, 1998.
- [48] A. McCabe, "Hidden Markov Modelling with simple directional features for effective and efficient handwriting verification," in *Proceedings of the Sixth Pacific Rim International Conference on Artificial Intelligence, Melbourne, Australia, 2000*.
- [49] D. Mital and K. Lau, "A microprocessor-based signature verification system," *IEEE Transactions on Consumer Electronics*, vol. 35, no. 4, pp. 845–51, 1989.
- [50] M. Munich and P. Perona, "Visual signature verification using affine arc-length," *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 180–6, 1999.
- [51] V. Nalwa, "Automatic on-line signature verification," *Proceedings of the IEEE*, vol. 85, no. 2, pp. 215–39, 1997.
- [52] W. Nelson, W. Turin, and T. Hastie, "Statistical methods for on-line signature verification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 749–70, 1994.
- [53] 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, pp. 710–17, 1990.

- [54] M. Paulik, N. Mohankrishnan, and M. Nikiforuk, "A time varying vector autoregressive model for signature verification," *Proceedings of the 37th Midwest Symposium on Circuits and Systems*, vol. 2, pp. 1395-8, 1994.
- [55] R. Plamondon, "The design of an on-line signature verification system: From theory to practice," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 3, pp. 795-811, 1994.
- [56] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-285, 1989.
- [57] L. Rabiner, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.
- [58] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, no. 1, 1978.
- [59] C. Schmidt, "Signature verification using time-delay neural networks," *Proceedings of the 37th Midwest Symposium on Circuits and Systems*, vol. 2, pp. 1395-8, 1994.
- [60] E. Singer and R. Lippmann, "A speech recognizer using radial basis function neural networks in an HMM framework," *IEEE Proceedings*, 1992.
- [61] J. Sternberg, "Automated signature verification using handwriting pressure," *WESCON Technical Papers*, vol. 31, no. 4, 1975.
- [62] L. Tseng and T. Huang, "An online Chinese signature verification scheme based on the ART1 neural network," *IJCNN International Joint Conference on Neural Networks*, vol. 3, pp. 624-30, 1992.
- [63] S. K. V. Valtchev and S. Young, "Recurrent input transformations for Hidden Markov Models," *ICASSP*, 1993.
- [64] C. Wen, M. Chang, B. Jeng, and H. Yau, "Signature verification based on distortion measure and spectral correlation," *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 2564, pp. 252-60, 1995.
- [65] T. Wessels and C. Omlin, "A hybrid system for signature verification," *Proceedings International Joint Conference on Artificial Neural Networks*, vol. 5, pp. 509-14, 2000.

- [66] A. Wilson and A. Bobick, "Nonlinear parametric Hidden Markov Models," *MIT Media Lab Perceptual Computing Section Technical Report 424*, 1996.
- [67] Q. Wu, S. Lee, and I. Jou, "On-line signature verification based on logarithmic spectrum," *Pattern Recognition*, vol. 31, no. 12, pp. 1865-71, 1998.
- [68] Q. Wu, S. Lee, and I. Jou, "On-line signature verification using LPC cepstrum and neural networks," *IEEE Transactions on Systems, Man and Cybernetics - Part B*, vol. 27, no. 1, pp. 148-53, 1997.
- [69] Q. Wu, S. Lee, and I. Jou, "Online signature verification based on split-and-merge matching mechanism," *Pattern Recognition Letters*, vol. 18, no. 7, pp. 665-73, 1997.
- [70] Y. Xuhua, T. Furuhashi, K. Obata, and Y. Uchikawa, "Selection of features for signature verification using the genetic algorithm," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 1037-45, 1996.
- [71] L. Yang, B. Widjaja, and R. Prasad, "Application of Hidden Markov Models for signature verification," *Pattern Recognition*, vol. 28, no. 2, pp. 161-70, 1995.
- [72] M. Yasuhara and M. Oka, "Signature verification experiment based on nonlinear time alignment: A feasibility study," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 3, pp. 212-6, 1977.
- [73] G. Zavaliagos, Y. Zhao, J. Makhoul, and R. Schwartz, "A hybrid segmental neural net/Hidden Markov Model system for continuous speech recognition," *IEEE Proceedings*, 1994.