ABSTRACT

RIGGAN, BENJAMIN S. Recognition of Sketch-based Passwords with Biometric Information Using A Generalized Simple K-Space Model. (Under the direction of Wesley E. Snyder.)

For many decades, text-based passwords have been the norm for information security. Recently, many have proposed to replace traditional passwords with various types of graphical passwords, such as drawing-based, recognition-based, cue-point based systems. In this work, one such drawing-based, or drawmetric, approach is presented. This proposed method combines both spatial and biometric information at a low level in order to achieve both security and usability (in terms of recognition performance).

The proposed algorithm for recognizing sketch-based passwords is an extension and generalization of a shape recognition algorithm called Simple K-Space (SKS). SKS achieves a sufficient level of performance by accumulating similarity evidence. If enough evidence exists, then the sketches are considered to be similar and access is granted. Otherwise, system access is denied due to insufficient proof of similarity. The algorithm has several advantages over existing methods, including invariance to similarity transformations, robustness to starting and ending points, and fusion of biometric information at a low level.

The novel authentication system, which uses the generalized SKS algorithm, is a semi-supervised system that requires minimal training (3–5 examples per user) during enrollment. While other approaches, such as support vector machines (SVMs), artificial neural networks (ANNs), and hidden Markov models (HMMs), require many exemplars in order to achieve sufficient performance, the generalized SKS approach performs extremely well with few training samples and minimal supervision.

In this work, a new theoretical perspective for sketch-based passwords is provided, which is validated using both synthetic and hand drawn datasets (e.g. DooDB and BioS-

ketch databases). The new insights and observations provided include uniqueness of the model, tolerance (or robustness) of the framework, and computational efficiency. Furthermore, several experiments are conducted in order to compare performances, study variability due to the human-computer interaction, and provide benchmark performances.

The experimental results show that adding pressure as a biometric feature to the SKS model representation reduces the equal error rate (EER)—rate at which the number of false acceptances and false rejections are equal—by more than 12%. The authentication system is also compared with both the Fréchet distance and dynamic time warping (DTW). SKS outperforms Fréchet with more than 7% reduction in EER using random forgeries and more than 17% reduction using skilled forgeries, and it performs as well (if not slightly better) than the state-of-the-art DTW method. Additionally, a human factors study is performed in order to demonstrate the variability in sketch-based passwords. Despite the known variations from the human-computer interaction process, our findings show that, in general, users are sufficiently capable of reproducing a draw with associated biometric with sufficient accuracy and precision.

Recognition of Sketch-based Passwords with Biometric Information
Using A Generalized Simple K-Space Model

by
Benjamin S. Riggan

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Electrical Engineering

Raleigh, North Carolina

2014

APPROVED BY:

H. J. Trussell	Edgar Lobaton
Larry K. Norris	Cliff Wang

Wesley E. Snyder
Chair of Advisory Committee

DEDICATION

To my beloved wife and my family who have all encouraged and motivated me along way

BIOGRAPHY

Benjamin S. Riggan was born in Charleston, S.C., but grew up outside of Raleigh, N.C. From an early age, he exhibited success in math and science. He attended North Carolina State University, first receiving a BS degree in Computer Engineering in December of 2009. While working as an undergraduate teaching assistant, he soon became interested in the fields of computer vision and pattern recognition. He worked on a autonomous vehicle research project headed by his future advisor and mentor, Dr. Wesley E. Snyder.

Continuing his education, Benjamin received his MS degree in Electrical Engineering from North Carolina State University in December of 2011. During his master's, he studied under the direction of Dr. Wesley E. Snyder. He began working in medical imaging, studying how to detect microcalcifications in mammograms using machine learning and pattern recognition techniques.

His dedication and success under Dr. Snyder, led to funding for PhD from the US Army Research Office. Therefore, continuing his studies under the direction of Dr. Snyder, he will receive his PhD in Electrical Engineering from North Carolina State University in the fall of 2014.

In August of 2014, Benjamin plans on working for the US Army Research Lab as postdoctoral researcher. He will continue researching in the fields of computer vision and pattern recognition, and hopefully will build a successful career in this area of research. His goals for the future include lifetime learning, state-of-the-art research, and mentoring/teaching future leaders.

ACKNOWLEDGEMENTS

I would like to thank committee for their guidance and critical feedback. First, I want to thank my advisor, mentor, and teacher Dr. Wesley Snyder for the time, effort, and dedication he invested into my education and my future. Words cannot express the appreciation I have for the countless lectures, discusses, and even intellectual debates throughout the years. I am truly thankful for the opportunities he gave me. Secondly, I would also like thank Dr. Larry Norris for the many discussions we regarding the mathematical theory surrounding the work in this dissertation. Although, I did not always see where I was going, he was there to point me in the right direction; letting me discover the answers I was seeking on my own. Next, I want to thank Dr. Joel Trussell who has always has inspired me to become a better researcher and instructor. His door almost always open, and he enjoys discussing difficult and challenging problems. Over the years, he has taught me how to provide better self-evaluations of my own work, for the purposes of improvement. I also thank Dr. Edgar Lobaton for his incites and perspectives. He helped me take a step back and see problems in a new light. Last, but not least, I would like to thank Cliff Wang from the US Army Research Office without whom I may have never had the opportunity for pursuing a PhD. I would also like to the thank him for the many career advancement opportunities that he has provided me.

I would like to thank all past and present member of the imaging group directed by Dr. Snyder, who have helped me along the way. I would like to thank Dr. Rachana Gupta, Yongqiang “Garfield” Huang, Theju Jacob, and countless other that have been there for intellectual discussions and guidance over the years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

$\mathbf{A}$	transition probability matrix for hidden markov models
a_{ij}	individual transition probability in $\mathbf{A}$
$\boldsymbol{\alpha}$	shape, curve, or sketch
$\boldsymbol{\alpha}^*$	unnormalized sketch
$\boldsymbol{\alpha}_d$	a discretely sampled sketch
$\boldsymbol{\alpha}_d^{(i)}$	a discrete sketch sample
$\boldsymbol{\alpha}_\ell$	a login sketch
$\boldsymbol{\alpha}_t$	a test shape or curve
$\text{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_t}$	Simple K-Space accumulator
$\mathbf{B}$	observation probability matrix for hidden markov models
$\mathscr{B}$	domain interval for pressure
b or $b_{\boldsymbol{\alpha}}$	local drawing pressure
b_{jk}	individual observation probability in $\mathbf{B}$
C or C'	covariance matrix (appendix only)
$\mathscr{C}$	a contour
c	class label for hidden markov models
D	cumulative dissimilarity (DTW), or a set of input observations (RANSAC)
d	point dissimilarity (DTW), or number of dimensions (other contexts)
d_F	Fréchet distance
$\delta(\cdot)$	a delta function
$\mathbf{E}$ or $\mathbf{E}^*$	eigenvector (appendix only)
$\mathbf{e}$ or $\mathbf{e}^*$	eigenvector (appendix only)
ϵ	threshold (DTW)
ε	margin of error for inlier (outlier) detection
$\mathscr{F}$	feature space (generalized SKS)
F_i	Fourier coefficient
f	usually denotes an arbitrary function (except when denoting frequency)
$f_{\boldsymbol{\alpha}}^{(i)}$	feature (generalized SKS)
$\mathscr{G}$	high dimensional feature space (appendix only)
g	usually denotes an arbitrary function
h	smoothing parameter for kernels and density estimators
h_2	differential entropy (in bits)
I	set of inliers (RANSAC)
i	usually used as an index
j	usually used as an index, but in the context of complex numbers $j = \sqrt{-1}$
K	a kernel function
$\mathscr{K}$	domain interval for curvature
k	usually used as an index

κ or $\kappa_{\boldsymbol{\alpha}}$	local curvature of shape, curve, or sketch
κ_{max}	maximum curvature
L	SKS likelihood measure
L_{sym}	a *symmetric* SKS likelihood measure
l	number of features (DTW)
ℓ_f or ℓ_b	used to denote forward and backward bisecting lines for curvature estimation
λ	eigenvalue
M	used to denote a number of elements
$\mathscr{M}$	space of models
m	used to denote a number of elements
$m_{\boldsymbol{\alpha}}$	model (SKS)
$m_{\boldsymbol{\alpha}}^{(i)}$	a discrete model sample
$\tilde{m}_{\boldsymbol{\alpha}}$	unnormalized model (SKS)
N	used to denote a number of elements
$\mathscr{N}_p$	size of local neighborhood around p
n	used to denote a number of elements
$\mathscr{O}$	Big-O notation
o_i	observation symbol for hidden markov models
Ω	correspondence map or warping path (DTW)
ω_i	hidden markov model state
P	set of model parameters (RANSAC)
$\mathscr{P}$	space of sketches
p	number of sample points
$p(\cdot)$	probability density function
$\mathbf{p}$	arbitrary path
Φ	non-linear mapping (appendix only)
φ or φ_i	model (appendix only)
ψ	mapping from the "space of sketches" to the "space of models"
Q	random subset of observations (RANSAC)
q_i	denotes a points (used in context of curvature estimation)
$\mathbb{R}$	set of real numbers
r	radius
ρ or $\rho_{\boldsymbol{\alpha}}$	local Euclidean distance between shape, curve, or sketch and reference point (SKS)
ρ_{max}	maximum Euclidean distance
$\mathbf{S}$	signature sequence
$\mathscr{S}$	domain interval for arc length
$\mathbf{SO}(n)$	special orthogonal group
s or $s_{\boldsymbol{\alpha}}$	arc length

$\mathbf{s}_i$	local signature feature vector
$\mathbf{\Sigma}$	covariance matrix
σ^2_{ρ}	variance corresponding to the Euclidean distance feature
σ^2_{κ}	variance corresponding to the curvature feature
$\sigma^2_{f^{(i)}}$	variance corresponding to an arbitrary feature
T	period (related to frequency)
$\mathbf{T}$	signature sequence
T_{iso}	isometric transformation
t	time
$\mathbf{t}_i$	local signature feature vector
τ	dummy arc length variable
Θ	domain interval for drawing direction
θ or $\theta_{\boldsymbol{\alpha}}$	local drawing direction for a sketch
$\mathbf{U}$	random variable (used entropy definition)
$\mathscr{U}$	domain interval for Euclidean distance feature
u	variable of integration
$\mathbf{v}$ or $\mathbf{v}_{\boldsymbol{\alpha}}$	feature vector (generalized SKS)
$\mathbf{v}_i$	discrete feature vector for a discrete sample point of a sketch
w_i	used to denote a weight factor
x	x coordinate
$x'(t)$ or $x'(s)$	first x derivative with respect to the parameter t or s respectively
$x''(t)$ or $x''(s)$	second x derivative with respect to the parameter t or s respectively
$\mathbf{x}$	vector representing spatial coordinates
$\mathbf{x}^*$	spatial location of peak in the accumulator (SKS)
y	y coordinate
$y'(t)$ or $y'(s)$	first y derivative with respect to the parameter t or s respectively
$y''(t)$ or $y''(s)$	second y derivative with respect to the parameter t or s respectively
x_i	x coordinate of a sample point
y_i	x coordinate of a sample point
z	normalization factor (SKS)
z_i	complex number $x_i + j\,y_i$
ζ	threshold (RANSAC)

CHAPTER 1

Introduction

There are many types of authentication systems, all of which may be classified as one of the following types:

1. token-based
2. knowledge-based
3. biometric-based

These systems are based on the following queries: *what a user a carries, what a user knows*, and *who a user is* respectively. Examples of token-based authentication include: keys, RFID fobs, and ID cards; knowledge-based examples are passwords and PINs; and biometric-based systems use face, fingerprint, voice, and signature recognition.

The most prevalent mechanism used for user authentication is text-based, or alphanumeric, passwords.

A major problem with protecting private information is the assumption that text-based passwords provide a sufficient amount of security. The truth is that more than 3,000 U.S. businesses were victims to cyberattacks in 2013 [74], most of which use some form of alphanumeric password.

In theory, traditional passwords provide a sufficient level of security. However, this assumes that the user has created a sufficiently complex password that adheres to specific requirements, such as being random, non-dictionary words, 8–15 characters in length, and including lowercase and uppercase letters, numbers, and special symbols. Text-based passwords are typically only useful against brute force attacks because the *password space*—the total number of possible passwords—is sufficiently large. Other attacks, such as phishing, spoofing, and shoulder surfing, easily overcome the security measures of text-based passwords.

There are two unanticipated consequences when using text-based passwords: 1) users completely ignore the security measures, or 2) users breach the security measures (despite adhering to password strength requirements). The first case is typical of many who ignore password protocols whenever possible. Users usually do this because they have one (or two) typical passwords that are easy to remember, but inadvertently insecure. The second scenario is typical of a user that creates a sufficiently complex password, but records his/her username and password either on paper or in a textfile called "passwords." Therefore, traditional passwords are not necessarily as secure as they claim.

In recent years, there have been many advancements in the type and number of user authentication methods, including gestures and patterns, picture-based passwords, and biometric recognition schemes. Each approach attempts to optimally balance *security*—preventing unauthorized users from accessing the system—and *usability*—granting access to authorized users with ease. The two extremes for such systems are: 1) perfect security,

which does not grant access to anyone, and 2) perfect usability, which grants access to all users. Neither case is desired. Ideally, the only person capable of gaining access should be the genuine user.

1.1 Sketch-based Passwords

In this dissertation, a new authentication approach, which combines a sketch (or drawing) with biometric information as the form of authentication, is presented. A sketch is obtained using a digitizer, tablet computer, or any device with a touch sensitive area for user input. Using a stylus (or finger) a person draws his/her sketch in order to access a secured system, which is either local to the device itself or on a remote system elsewhere.

The advantages of sketches are the following:

- memorability
- security
- reliability
- low cost

Sketches are easier to remember than character strings because the majority of people tend to prefer shapes and visual patterns over alphanumerics. They also offer better security through use of biometrics. With biometric properties, a system is better protected against brute force, shoulder surfing, and spoofing attacks. The exponential increase in the number of tablets and touch sensitive devices over the last decade has made the reliability of such a system possible because various types of recognition software on these devices are rapidly improving performance while the hardware is becoming more accurate and efficient. These

devices are also becoming more affordable, therefore keeping software development costs down.

The randomness due the human-computer interaction makes using a sketch-based system somewhat difficult. There are at least two questions to be answered: How accurate can a user replicate his/her own sketch? How secure are such sketch-based systems? The method used to compare sketches must be robust enough to deal with natural variation from the genuine user and simultaneously be secure enough to reject any forgeries. Thus, there are usability and security concerns with sketch-based passwords, as there are with other authentication systems.

The proposed system utilizes concepts from shape recognition, signature recognition, and biometric systems in order to form a robust and reliable authentication system. This method attempts to alleviate many of the aforementioned concerns. To our knowledge, no other sketch-based or drawing-based system achieves the same performance level as the authentication method proposed in this text.

1.2 Contributions

The main contribution of this dissertation is the authentication system itself, which includes the generalized framework for recognizing sketches with biometric information. Using this framework, additional theory is provided in the application of sketch-based passwords, including: computational complexity and the security/usability tradeoff. Additionally, we demonstrate that sketch-based passwords are an effective means for authenticating users by performing experiments with both synthetic and hand-drawn datasets, including a new general database referred to as the “Biometric Sketch” database or “BioSketch” database. We successfully demonstrate the improvement that one biometric, namely pressure, makes

when used with sketch-based passwords. Lastly, we study the human-computer interaction process for sketch-based passwords.

1.3 Summary of Results

The major results presented in the latter chapters of this work are the following:

1. 0.0% equal error rate (EER) on a set of synthetic shapes
2. 1.4% EER (random forgeries) and 28.0% EER (skilled forgeries) on "doodle" or sketches from the DooDB database [68] *without biometrics*
3. 1.6% EER (random forgeries) and 23.3% EER (skilled forgeries) on pseudo-signatures from the DooDB database [68] *without biometrics*
4. 0.0% EER (random forgeries) and 3.3% EER (skilled forgeries) on a set of hand drawn sketches
5. 1.74% EER (random forgeries) and 16.75% EER (skilled forgeries) on the BioSketch database

The terms *random* and *skilled* forgeries will be discussed in detail in Section 7.1.2.

These results demonstrate the effectiveness of the approach on a variety of datasets, multiple users (genuine and forgers), and under many conditions. Refer to the main text for complete details regarding each experiment.

1.4 Organization

The remainder of this dissertation is organized in the following manner.

Chapter 2 provides a summary of major literary works related to sketch-based passwords. This literature review provides an overview of handwriting recognition, alternative graphical-based passwords, and biometric systems. The topics discussed in this section provide a solid foundation for the remaining chapters.

Next, signature recognition (Chapter 3) and shape recognition (Chapter 4), which are the most relevant topics when discussing sketch-based passwords. In these two chapters, two approaches: Dynamic Time Warping (DTW) and Simple K-Space (SKS), which are state-of-the-art pattern recognition methods, are discussed in detail. Chapter 4 also introduces a generalized framework which is essential for the approach used for recognizing sketch-based passwords.

Chapter 5 introduces the novel sketch-based authentication system. This chapter is the core of the work presented in this dissertation. Here, the major components of the sketch-based password system are discussed. Additionally, new theory regarding the efficiency, security, and usability is proven, and experimental results are provided to support this theory.

Chapter 6 discusses many practical considerations with a discrete implementation, including sampling, feature estimation, and interpolation. Also, the memory usage versus speed tradeoff is discussed. Understanding this tradeoff is useful for knowing how best to implement a sketch-based password system on servers, desktop or laptop computers, and tablet computers.

Chapter 7 discusses the construction of the Biometric Sketch database or "BioSketch" database. Using sketches from the database, a human factors analysis is performed. This study aims to analyze the variability of sketches provided by genuine users and forgers (skilled and unskilled). Additionally, benchmark performances are provided for the database.

Lastly, the conclusions from this work are discussed and possible extensions are considered in Chapter 8.

CHAPTER 2

Background

2.1 Handwriting and Character Recognition

Handwriting recognition and handwritten character recognition are two topics that ultimately led to current research problems involving human-computer interactive processes, such as gesture, signature, and drawing recognition. Handwriting recognition consists of many different forms, including handwritten print or cursive [59, 70], different written languages (e.g. Chinese [64, 105], Arabic [40], and Latin [80]), characters [15] or numbers [23], and even context specific handwriting recognition (e.g. dates and checks [46]).

These two similar topics are categorized as either: *online* or *offline*. Online systems use a digitizer, such as a Wacom Intuos (Figure 2.1a), or tablet computer (Figure 2.1b) to electronically capture the handwriting. Online systems vary in both the type of data collected and the schemes used to capture information from a user. For example, the Wacom Intuous,

which is designed to be used with a laptop or desktop computer, uses a separate computer screen for user feedback; this may adversely affect the user experience. Other digitizers and tablet computers overlay the active surface (i.e. touch sensitive screen) on top of the screen for user feedback.

Offline systems, instead, use a camera or scanner to obtain digital image of the handwriting for recognition purposes. Offline systems are not only different from online systems in hardware but also in the type of features used. For instance, the dynamics of handwriting are usually unknown for offline systems.

(a) Wacom Intuos digitizer. Image obtained from [1]

(b) Samsung Galaxy Note 10.1 tablet computer. Image obtained from [2]

Figure 2.1: Example digitizer and tablet computer for online handwriting recognition

.

This work primarily focuses on online systems, however, the primary recognition components of offline systems are similar to those of online systems, after the handwriting is captured and preprocessed. Most handwriting and character recognition systems include the following stages: (1) *feature extraction* and (2) *classification*.

2.1.1 Feature Extraction

There are an innumerable number of features used for handwriting recognition; only a subset of features are introduced in this section. Many online systems use a Fourier descriptor as a set of features, and other offline/online systems use spatial or temporal features.

Fourier descriptors generally assume that a given contour is closed. Note that there are existing methods for closing contours [62], as well as extended Fourier descriptor methods for open contours [25].

There are multiple Fourier descriptors that may be used for character or handwriting recognition (see [65]), including:

- complex Fourier descriptor [71]
- affine-invariant Fourier descriptor [7]
- moment-invariant Fourier descriptor [10]

The complex Fourier descriptor is the only feature transform discussed since affine and moment invariant formulations use the same fundamental principle, except for the modification to achieve the desired invariances.

Let n sample points along a given contour $\mathscr{C}$ be represented by the complex[1] coordinates:

$$z_i = x_i + j y_i,$$

[1] In the context of complex numbers, $j = \sqrt{-1}$.

where the pair $(x_i,\ y_i)$ represents the planar coordinates the i^{th} contour sample.

Then, the descriptor is represented by the set of Fourier coefficients:

$$F_i = \sum_{k=0}^{n-1} z_k \exp\left(-\frac{j2\pi ik}{n}\right) \quad \text{for } i = 0 \ldots n-1.$$

Fourier descriptors not only provide a useful descriptor for handwriting recognition; they have a meaningful interpretation. The set of Fourier coefficients, $\{F_i\}_{i=0}^{n-1}$, describes components of the contour in the frequency domain. Fourier descriptors possess a relationship to transformations in the spatial domain, which are described as (see [97]):

- Scaling (in the spatial domain) produces Fourier coefficients that have all been multiplied by a constant
- Planar Rotation produces phase shifted Fourier coefficients
- Translation produces Fourier coefficients with a DC (i.e. zero frequency component) offset

These properties allow for invariance to rigid transformations of contours. For additional invariances, refer to [7, 10].

As mentioned previously, Fourier descriptors are not the only features used for handwriting recognition. Table 2.1 provides a variety of features that have been used in such applications.

After extracting a feature descriptor for the handwriting or characters, the next step is classification.

Table 2.1: List of local and global features used in handwriting applications.

Type	Feature	Description
Global	Time	The total time required to write [39, 53]
	Bounding Box	The total writing area required [58]
	Pen Ups	The total number of times a user lifts up the pen (or finger) while writing [39]
	Pen Downs	The total number of times a user presses the pen (or finger) tip down [39]
Local	Position	The x and y coordinates of each sample point, typically with respect to some reference [82]
	Displacement	The difference in x, Δx, and the difference in y, Δy, between sample points [50, 58]
	Velocity	The instantaneous rate of change in position with respect to time [53, 82]
	Acceleration	The instantaneous rate of change in velocity [43, 53, 82]
	Pressure	The point-wise amount of force applied at the pen (or finger) tip by the user in the process of writing [33, 43, 82]
	Curvature	The point-wise measure describing the amount of deviation from a line, which is commonly used to model shape [50]
	Azimuth	The angle between the pen's projection on the table and the tablet's coordinate reference angle [33]
	Altitude	The angle between the pen and the tablet [33]

2.1.2 Classification

Constructing a robust and invariant descriptor is one step, but recognizing or classifying them is another. Although there exist many different methods for recognizing handwriting using a feature representation, Hidden Markov Models (or HMMs) are commonly used for this purpose [3, 47, 48, 66].

The primary concepts for HMMs are briefly discussed here, and then a few HMM structures used in handwriting recognition applications are reviewed.

HMMs are widely used in both speech recognition [81] and handwriting recognition [3, 47, 48, 66] applications. A HMM is analogous to the *urn-and-ball* experiment [83], in which there are N urns (or vases) each containing M different colored balls. In this experiment, the urns are placed behind a curtain and a person chooses a sequence of urns, according to some random process, pulling one ball from each urn out from behind the curtain at time.

Similarly, a HMM has N *unobservable* or *hidden* states (where each state corresponds to an urn), which are denoted $\{\omega_1, \omega_2, \ldots, \omega_N\}$. The transition sequence between states is determine by a matrix of transition probabilities

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{pmatrix},$$

and the probability of observing a particular observation variable (or colored ball) from the set of possible observations,$\{o_1, o_2, \ldots, o_M\}$, is defined within each state. The probability of observing the k^{th} variable (or symbol) from the j^{th} state is denoted by b_{jk}. Thus, the set of

observation probabilities is denoted by the $N \times M$ matrix

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1M} \\ b_{21} & b_{22} & \cdots & b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \cdots & b_{NM} \end{pmatrix}$$

The urn-and-ball experiment provides a good illustration of a HMM, but how are they used for handwriting recognition (or any other pattern recognition problem)? There are two problems to consider: (1) the evaluation problem and (2) the learning problem.

The goal of the evaluation problem is simply to determine the probability that a given sequence of observations, **O**, was generated by the HMM, which assumes that the transition and observation probabilities, **A** and **B** respectively, are known. That is, the goal is to compute $Pr(c|\mathbf{O},\mathbf{A},\mathbf{B})$, which represents the probability that the observation belongs to some class c given observation and the HMM with known parameters **A** and **B**.

The purpose of the learning problem is to update the HMM probabilities, **A** and **B**, based on a collection of known observation sequences (presumably belonging to some class c). The only assumption for the learning problem is the basic structure—the number of states (and/or transitions) and observation symbols—of the HMM. The probabilities for the given HMM are updated using the *forward-backward* algorithm [30].

The forward-backward algorithm, unlike artificial neural networks (ANNs) and support vector machines (SVMs), does *not* seek an optimal solution for the HMM parameters, however, because this algorithm is a generalization of Expectation-Maximization (EM) an acceptably "good" solution may be obtained.

In general, the most common HMMs used in handwriting and character recognition are

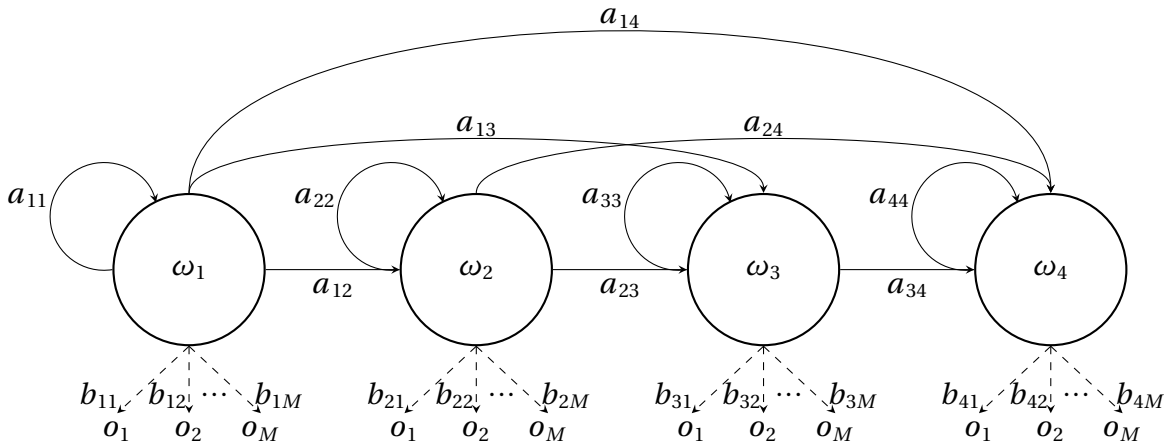


Figure 2.2: Example left-to-right HMM.

the *left-to-right* [47] HMMs (Figure 2.2), which prohibit any backward transitions (i.e. $a_{ij} = 0$ for $j < i$). There is also a subset of left-to-right HMMs that prevent the skipping of any state, meaning that from any single state, ω_i, there are only two possible transition: the same state (ω_i) or the next state (ω_{i+1}).

Many have used HMMs for handwriting recognition and achieved varying levels of performances. A summary of the different HMM approaches are presented in Table 2.2. The table shows that there are numerous ways to implement a HMM for handwriting recognition, including word, character, or subcharacter models and different training methodologies.

2.2 Graphical Passwords

Graphical passwords are not new (in fact the idea has been around for quite sometime), but there are still many questions regarding their security and reliability. Graphical passwords belong to one of the following taxa (according to[28]):

1. Drawmetric—recall-based methods

2. Cognometric—recognition-based methods (also referred to as search metric [84])

3. Locimetric—cued-recall based methods.

Table 2.2: Summary of various HMM approaches for handwriting and character recognition.

Author(s)	**Structure**	**Training**	**Performance**
Hu et al. [47]	left-to-right without state skipping	"nebulous stroke models"—subcharacter model (3100+ samples)	~3.1% error (~1600 test samples)
Rigoll et al. [88]	left-to-right without state skipping	character model (1640 samples)	0.65% error (single characters) and 30% error (word recognition)
Martin et al. [67]	left-to-right	character model with Baysian Information Criterion (BIC) for optimal number of states	6% error (discrete HMM) and 0% error (continuous HMM)

Drawmetric systems require users to construct a unique sketch or drawing as a password. This construction is referred to as the *enrollment* phase. Then, at *login* time genuine users are expected to recall and accurately reproduce the password. This type of system requires that a user physically draw a "password" using a digitizer or tablet, which is a noisy process itself. However, prior to drawing, the user must recall the sketch from memory (including the associated biometric properties, e.g., pressure). Then, the user must convert that depiction into a sequence of muscle movements to physically reproduce the drawing. Therefore, drawing a sketch-based password requires a complex human-computer interaction.

Although drawmetric system have been previously proposed, they have not (until now) been feasible because technologies, such as processors, memory systems, and touch screen in mobile devices, were not mature. The ubiquitousness of tablet computers in society paves

the way for success and acceptance of drawmetric application.

Probably, the most notable drawmetric system is Draw-A-Secret (DAS) [52]. The DAS approach uses a coarse grid, which is displayed to the user, to encode the drawing. The drawing is encoded using the grid cells. The ordered sequence of grid cells that the drawing enters produces a password string. Thus, the matching procedure for DAS is virtually no different from text-based passwords (except for the fact that the string is generated from a drawing); the string produced at login time is compared with the string stored during enrollment.

There are some extensions to DAS which have improved performance. For example, Background Draw-A-Secret [32] improves performance by adding background images. Yet Another Graphical Password (YAGP) [37] improves performance by using a finer grid than the original DAS approach. By removing the visible grid for drawing and including stroke color as an additional feature, Passdoodle [38, 99] improves performance.

One drawback to DAS and similar approaches is the difficulty of encoding near intersection points of grid cells. To alleviate this problem, the Pass-Go [98] approach uses grid intersection points as anchor points instead. Thus, the encoding records an ordered sequence of horizontal, vertical, and diagonal strokes.

Cognometric systems, instead of drawing, demand that users recognize and select the correct set of images from a randomly assorted set of images. These types of systems leverage the fact that humans tend to recognize something easier than they can remember something. For example, the human visual system is adept at recognizing faces and object under different lighting conditions, rigid transformations, and occlusions. Although cognometric systems still require the interaction between a person and a computer, in this case, the procedure is less complicated because the user is not required to produce a drawing. Instead, users are required only to indicate the recognition of a face, landscape, or any other image, using a

mouse, keyboard, or touch screen.

The best example of a recognition-based system is PassFaces [78]. During enrollment, the user is presented with a random set of faces and asked to select a subset of them as a "password." At login time, the user is presented with the correct set of faces dispersed among a random set of other faces. The user is then asked to identify the correct set of faces.

Instead of faces, the Déjà Vu [29] system uses a set of images with graffiti-like patterns and artwork. Studies [27, 85] have shown that, even with choosing faces, people are predictable in their choice of password. Predictability in security implies that the entropy, or measure of randomness that is related to "security," is reduced. Déjà Vu attempts to alleviate this problem by making the images very randomized, so that a bias or preference does not exist toward any particular image.

Locimetric systems, instead of requiring users to remember entire images, have users recall and identify specific points or regions within an image. The idea is to reduce the amount of information that a user is required to remember. Given a larger image, a user selects an ordered sequence of points or regions located within the image. In principle, the image should help stimulate the process of remembering the points previously chosen, hence the term cued-recall.

The most prominent cued-recall based method is PassPoints [100, 101, 102]. PassPoints presents the user with a single image—typically having numerous landmarks: people, buildings, objects, etc.—in which during enrollment the users is supposed to select 5 different locations. At login time, the user is again presented with the same image, and the user must chose the correct set of points within the image in precisely the same order.

An extension to the PassPoints philosophy, called Cued Click-Points [22, 21], presents the user with multiple images (one at a time). In each image the user selects only one point to use. Similarly, the ordered sequence of locations in each image becomes the password. If any

one point is incorrect, access is denied.

Many of the methods using drawmetric, cognometric, and locimetric password systems are compared and contrasted in [16]. This literature survey analyzes the multitude of methods in terms of both usability and security, especially the vulnerabilities to different attacks (e.g. brute force, phishing, and shoulder surfing).

2.3 Biometric Systems

Biometrics systems—systems that measure and analyze certain biological properties for the purposes of identity recognition [36]—are steadily becoming a part of everyday life, including in commercial access systems, personal computers, and smart phones. Biometrics are increasingly popular because they are supposed to be universal, distinct, permanent, and collectable [51] properties that uniquely identify an individual. For these reasons, biometrics are considered much more secure than traditional passwords.

Biometrics include biological properties such as fingerprints, voices, faces, and even handwriting. Fingerprints have been widely used and accepted in forensic, commercial, government security, and consumer applications. Despite the fact that fingerprints are known to be unique to every individual, there are still concerns in terms of a system's ability to recognize fingerprints (especially with consumer applications). For example, fingerprint recognition is very sensitive to noise and occlusion. In most cases, the signal-to-noise ratio (SNR) is low, which increases the difficulty of recognizing fingerprints accurately (implying the possibility of a breach in security).

Occlusion also presents a problem for fingerprint recognition because the fingerprint (and its features) must be visible (at least mostly visible) in order to determine with sufficient confidence that it matches another fingerprint. There are several reasons for occlusion of a

fingerprint including varying poses and pressure.

Voice recognition systems, although not as widely used in security application, are used applications such as voice-to-text in cars and smart phones, and voice command systems. There are many reliability concerns when it comes to voice recognition systems for user authentication. Multiple variables, such as distance to microphone(s), background noise, and vocal anomalies (e.g. colds, dialects, and accents), affect the reliability of a voice recognition system. The most concerning of these are the anomalies because they naturally occur to human voices, but voice recognition systems still experience difficulty in handling these effects.

Facial recognition systems for authenticated access are also becoming more common because of advancements with this technology. However, there are still concerns about the reliability of these systems. Facial recognition most commonly experiences difficulties with occlusion. Faces can be occluded because of clothing or accessories, lighting conditions, or even pose. As with fingerprints, the confidence of the face recognition is reduced when there is occlusion. Under normal operating conditions, one would expect facial recognition to perform quite well. However, there are legitimate security concerns with false acceptances—when the system grants access to an unauthorized person.

Although it is difficult, under normal conditions, to change or alter a person's fingerprint, voice, or face, it is certainly plausible for an authorized user to be coerced (with extreme force) into unlocking the system. Thus, granting access to unauthorized users. For example, a fingerprint can be reproduced (or in the extreme case a finger can be severed), a voice may be recorded or copied, and a face recognition system can be fooled with a picture of a face. Each requires going to great lengths, but all are reasonable means to fool a biometric system.

According to Jain et al., "foolproof personal recognition systems simply do not exist and perhaps, never will" [51]. However, in this dissertation, we attempt to introduce a new system

that bridges the gap between security and usability.

CHAPTER 3

Signature Recogntion

Before discussing the recognition problem associated with sketches, it is necessary to discuss the related problem of signature recognition. Signatures may be viewed as a special type of sketch; one with high complexity and high intra-class variance. This special case is one that has been widely studied.

Multiple methods, including dynamic time warping (DTW) [91] and continuous dynamic time warping (CDTW) [73, 93, 94], regional correlation [43, 44, 63], and local shape-based methods [75], have been used to solve the problem. DTW and CDTW are warping methods, which align one signature with another in order to determine similarity. This is done by means of a dynamic programming [11] approach and setting up an optimization problem to solve the correspondence problem. DTW works by optimally aligning discrete points on the signatures. However, CDTW allows a discrete point on one signature to correspond to a continuous point between two samples on another signature. The warping process is performed

in a way such that the dissimilarity between two signatures is minimized. After optimizing, if the dissimilarity is sufficiently small then the signatures are considered to be the same. Regional correlation approaches to signature recognition also look for correspondences. Instead of using singular points on the signature (like DTW), local regions (i.e. segments) are used to determine correspondence. Using a correlation measure, the cross-correlation of two signatures results in a correlation for each segment. Then, an average correlation is used to determine signature similarity.

Machine learning techniques, like hidden Markov models (HMMs) [30, 103] and artificial neural networks (ANNs) [57, 58] have also been applied to signature recognition. HMMs approach signature recognition from the random process perspective—the goal is to construct a stochastic model capable of generating class of signatures with high probability. A HMM is a fixed state machine with some number of states, transitions, and observations (from a defined vocabulary). Using a set of known signatures from the same class, the model can be trained. By training, it is implied that the state transition probabilities are iteratively updated (with the hope of generalizing the model). Then, a signature from an unknown class is compared against the model, by determining the likelihood that the given HMM (with known probabilities) will generate that particular signature. If the likelihood is sufficiently large (which implies some type of thresholding), then the signature belongs to the same class as the ones used to train the model.

Similarly, ANNs use a set of signatures to train model. Instead of using explicit probabilities, ANNs learn (i.e. optimize) the synaptic weights of the network using *back propagation*. In general, ANNs are capable of learning or recognizing signatures, but generalization of these networks usually depend on a priori assumptions about the type, structure, and number of connections. Although, probability distributions are not explicitly computed, distributions are implicit in the sample space. Therefore, performance of either HMMs or ANNs directly

depends on the assumptions used.

3.1 The Correspondence Problem

A common problem that many signature recognition method attempt to solve is the *correspondence problem*—the problem of finding corresponding or matching points, typically, in presence of noise and arbitrary transformations (temporal and spatial). In most cases, the correspondence problem refers to determining which regions of one image match those of a second image, where the second image is produced from the spatial displacement of the camera or objects in the field of view or a time lapse. However, for signature recognition, this problem means determining which points/regions of the signature match those of another signature.

Under ideal conditions, solving the correspondence problem is trivial. However, under general deformations and noise caused by a random process, e.g. a signature or handwriting process, finding correspondences proves to be more difficult. This is because under severe perturbations the unique features or properties of a signature may appear to vanish. Thus, the detection of correspondences is more difficult.

One approach that attempts to find an optimal alignment (i.e. correspondence) of two signatures for comparison is DTW. This method is discussed in more detail below.

3.2 Dynamic Time Warping

The goal of DTW is to the find the correspondence map, Ω, which can be graphed using a grid (Figure 3.1), that minimizes the total dissimilarity between two signatures, $\mathbf{S} = [\mathbf{s}_1\ \mathbf{s}_2\ \cdots\ \mathbf{s}_n]$ (an $l \times n$ matrix where l is the number of features and n the number of samples) and

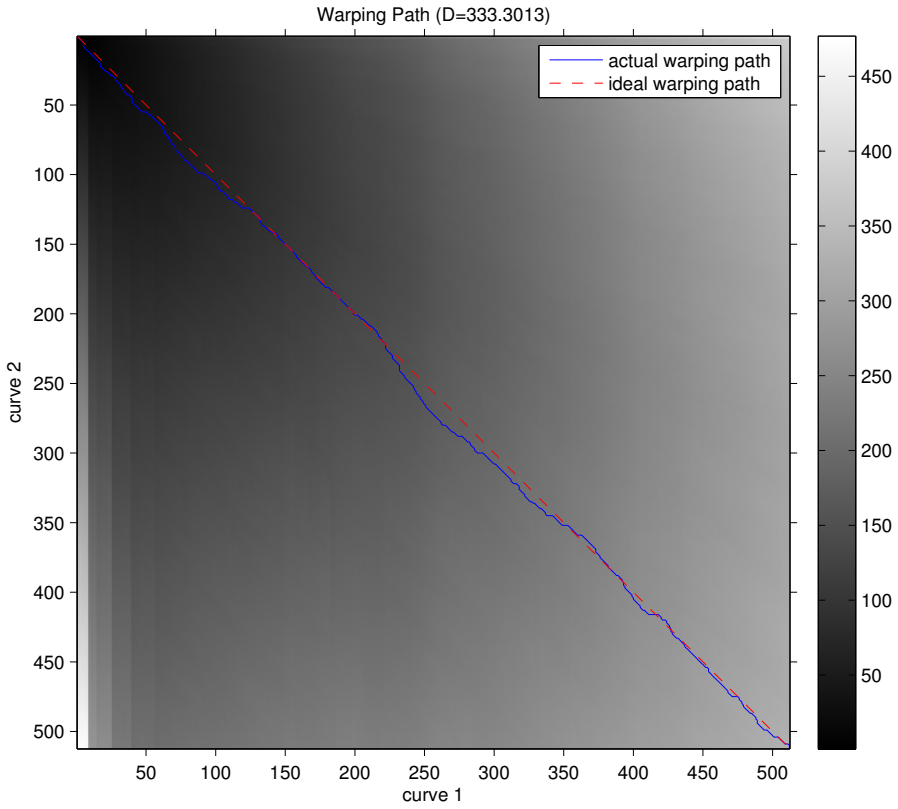


Figure 3.1: The optimal warping path for two signatures are plotted. The *dashed* line shows the ideal alignment, and the *solid* line shows the actual alignment of the two signatures. The grayscale background at a point $(i,\ j)$ shows cumulative dissimilarity, $D(\mathbf{s}_i, \mathbf{t}_j)$, which represents the total dissimilarity up to aligning the i^{th} sample point of $\mathbf{S}$ and j^{th} sample of $\mathbf{T}$. Notice that the cumulative dissimilarity increases as the path goes away from the upper left corner of the image, which is expected for two *distinct* signatures.

$\mathbf{T} = [\mathbf{t}_1\ \mathbf{t}_2\ \cdots\ \mathbf{t}_m]$ (an $l \times m$ matrix). The cumulative dissimilarity by aligning the i^{th} point of $\mathbf{S}$ with the j^{th} point of $\mathbf{T}$ is

$$D(\mathbf{s}_i, \mathbf{t}_j) = \min_{\alpha} \{D(\mathbf{s}_{i-1}, \mathbf{t}_\alpha)\} + d(\mathbf{s}_i, \mathbf{t}_j) \tag{3.1}$$

where $d(\mathbf{s}_i, \mathbf{t}_j)$ is the additional cost associated with aligning samples i and j (usually a function of $\|\mathbf{s}_i - \mathbf{t}_j\|$ where $\|\cdot\|$ is the L_2 norm). The first term on the right hand side of

Equation 3.1 is minimizing the cumulative dissimilarity, over all α such that the point $\mathbf{t}_\alpha$ aligns with the point $\mathbf{s}_{i-1}$. Using recursion, and thus determining the set of corresponding points (not necessarily one to one correspondence), the cumulative dissimilarity between the signatures is then determined by the value $D(\mathbf{s}_n, \mathbf{t}_m)$. Note, that this dissimilarity has never been shown to be a metric because no proof exists that it satisfies the triangle inequality. All other properties required to be a metric are satisfied. However, in [20] it is shown that the triangle inequality is not violated in a large (on the order of 15 million) set of speech patterns. Therefore, if DTW is shown with statistical validation to *loosely* satisfy the triangle inequality then it can approximately be considered as a metric.

Two signatures are considered to match if and only if $D(\mathbf{s}_n, \mathbf{t}_m) < \epsilon$, where ϵ is some experimentally determined threshold on the total dissimilarity. Otherwise, the signatures are too different to be considered a match.

The algorithmic complexity of DTW is $O(n^2)$ [39]. However, this is typically a worst case scenario. This complexity is alleviated some by use of global and local constraints. Typical constraints include:

Boundary Conditions Typical boundary constraints include the alignment of $\mathbf{s}_1$ with $\mathbf{t}_1$, and $\mathbf{s}_n$ with $\mathbf{t}_m$.

Monotonic Conditions Assuming the i^{th} and j^{th} points are aligned, then point $i+1$ must be aligned with a point $k \geq j$.

Global Path Conditions This restricts the warping path from going to far away from the ideal path.

Slope Weighting Assuming certain slopes (i.e. derivative approximation or "change" in samples from one signature over "change" in samples from the other signature) between

two points on the correspondence path are desired over others, then the slope weights are added in the cost function.

3.3 Experiments and Results

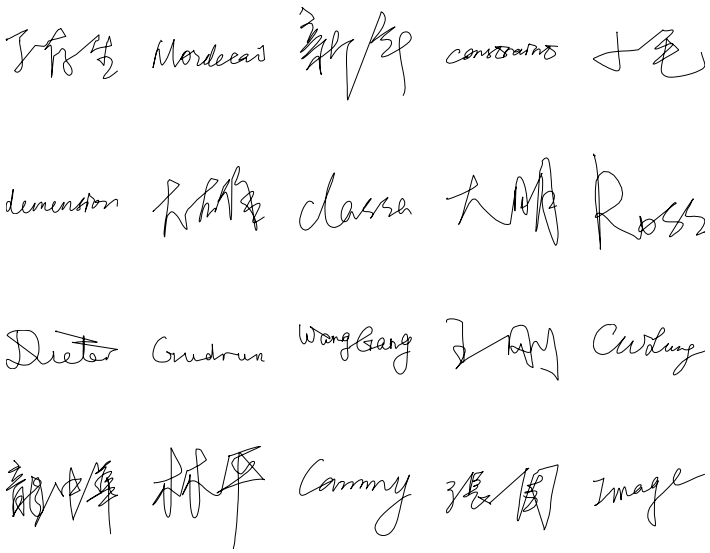
In this section, an experiment using DTW for signature recognition is conducted, and the corresponding results are reported.

The *Signature Verification Competition* (SVC2004) databases [104] are used to test a DTW algorithm. Examples from each database are shown in Figure 3.2. Both databases include 20 genuine signatures and 20 forgery signatures for 40 different individuals. This experiment examines only the skilled forgery case because it is the more difficult and interesting situation.

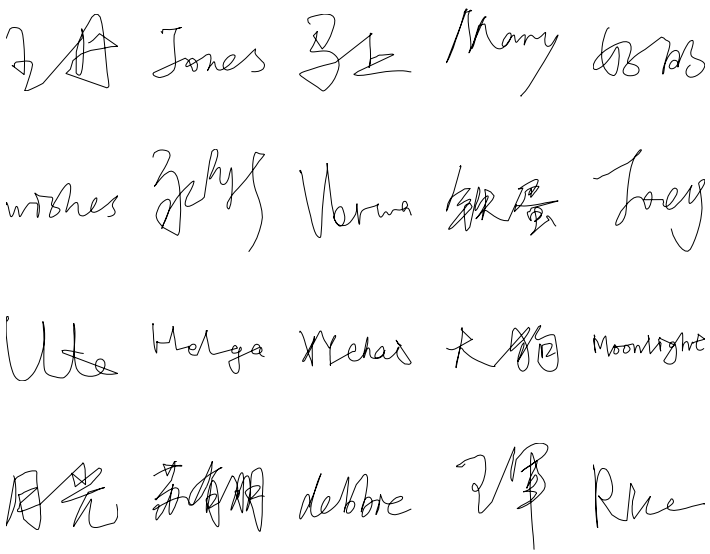
The experiment is as follows:

For every user, each genuine signature is compared (using DTW) to every other signature for that user (both genuine and forgery). If a genuine signature is considered too dissimilar, then it is counted as a false rejection. Likewise, if a forgery is considered similar enough, then it is counted as a false acceptance. Performance for most signature recognition (and biometric systems) is measured by the false acceptance rate (FAR) versus false rejection rate (FRR) curves. In some cases, performance is reported in terms of the equal error rate (EER), which is the rate where the FAR and FRR are equal.

Figure 3.3 shows the FRR plotted against the FAR for both sets of signatures. The EERs achieved are 0.67% and 1.31% for each database respectively.



(a) Database 1 SVC2004.



(b) Database 2 SVC2004.

Figure 3.2: Example signatures from SVC databases.

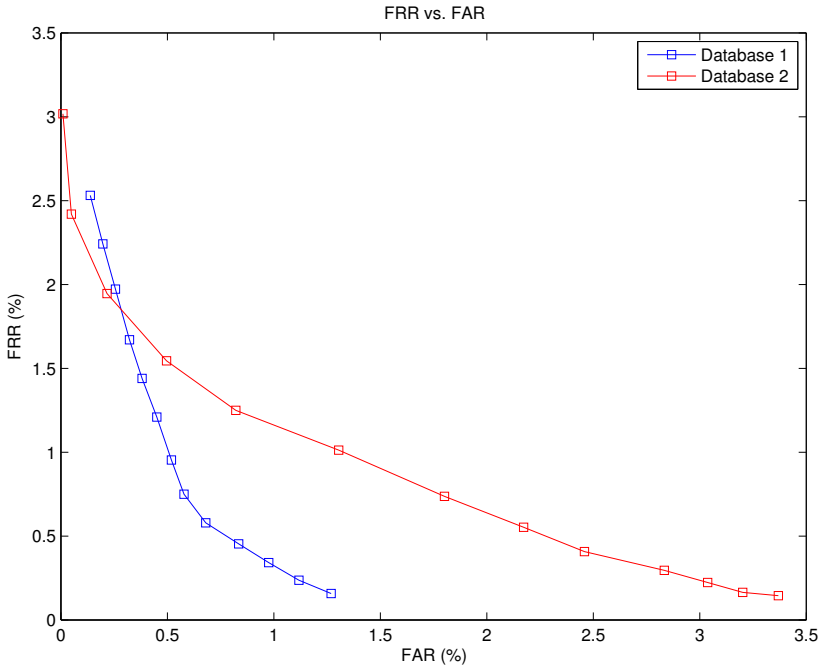


Figure 3.3: Performance curves for both SVC2004 databases.

3.4 Discussion and Analysis

The extension of DTW to recognize sketches (rather than signatures) is not a difficult one. However, there are two potential problems with the DTW approach: 1) start/end point dependence and 2) efficiency/generalizability. The assumption that a user starts and ends at the same *exact* locations when drawing a sketch (or signature) is not a good one. DTW assumes that the starting point of a sketch is best aligned with the starting point of another sketch, and likewise the end point of a sketch is best aligned with the end point of another sketch. However, this is frequently not the case due to user variability in start/end points.

Secondly, DTW only describes the comparison between two sketches (or signatures). When deciding if a sketch (or signature) belongs to a class of sketches (or signatures), then usually DTW has to make to compare the testing sketch/signature with multiple (if no all)

sketches (or signatures) from that class. Alternatively, it is more efficient to have a single template or reference sketch, which is composed of multiple exemplars, for the purposes of comparison.

CHAPTER 4

Shape Recognition

4.1 What is Shape?

Probably the most distinctive difference between two dissimilar sketches is captured by their shape. The earliest studies of shape theory are found in [18, 19, 54, 108]. These early works led to Kendall's shape space. Kendall defines shape to be "what is left when the differences which can be attributed to translations, rotations, and dilatations have been quotiented out" [55]. Much research has been performed in the area of shape recognition and shape matching. There are many varied approaches used for differentiating the detailed differences between shapes, including (but not restricted to):

1. Fourier Descriptor methods
2. Statistical methods
3. Accumulation methods

4. Shape matrix and other methods

4.2 Fourier Descriptors

Fourier descriptor methods, such as [8, 79, 106], are appealing for matching curves and shapes because of the invariances to similarity transformations: translation, rotation, and scale. Also, Fourier descriptors exhibit a useful inverse property in which a shape may be reconstructed from the descriptor. The complete descriptor has an infinite number of coefficients, but truncating to a sufficiently large number of coefficients produces minimal error between the original curve and the reconstructed curve. As the number coefficients is reduced, the corresponding curve becomes smoother, and less similar to the original.

4.3 Statistical Methods

Many statistical shape models exist for shape recognition, including methods using kernel density estimation [89], principle components analysis (PCA) [97], kernel PCA (or KPCA) [92], and shape context [13]. Density estimators use a probabilistic representation of shape, and the similarity between shapes is determined by likeness of the distributions. An approximation of the distribution is determined from a collection of points, $\{\mathbf{x}_i\}_{i=1}^n$, which are unordered since shapes do not explicitly have any parameterization. Intuitively, these points are blurred using a kernel (e.g. Gaussian) [95]

$$K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right).$$

The kernel used to estimate the shape distribution usually includes some form of a smoothing parameter, h, that provides a trade-off between accuracy and smoothness of the distribution.

PCA (KPCA) is a method which computes the dominant linear (nonlinear) directions of maximum variance. PCA methods provide a compact and invariant representation of shape, which are typically used to discriminate dissimilar shapes. PCA works by determining a different coordinate system, which is more representative of the shape distribution. This coordinate system is defined by a set of orthonormal axes that are determined by the principle eigenvectors of the data covariance matrix. KPCA also defines a set of orthonormal axes in order to define a new coordinate system. However, the difference is that KPCA does not operate in the input space (like PCA). Instead, the input is mapped nonlinearly to higher dimensional space, and classic PCA is performed in this space. Therefore, when the orthonormal axes in the higher dimensional space are mapped back into the input space, the principle directions are nonlinear. In either case, the coordinate system is used to construct a more compact representation which can discriminate dissimilar shapes. For more detail regarding PCA and KPCA, refer to Appendix B.

The shape context approach, which usually assume a parameterization or ordering of points, solves the shape recognition problem using a histogram approach. This approach computes the distances and angles (normalized by the mean) between every pair of adjacent points, and then a histogram is constructed. Note that the histogram bins are constructed such that points closer together (in terms of distance) are more discriminative. A histogram, as described, is constructed for any two shapes. Then, shape matching is performed by comparing the similarity between the histograms using the Chi-squared similarity measure. A summary of statistical shape methods is provided in [26].

4.4 Accumulation Methods

Accumulation methods, like the Hough Transform [9], have been applied to various shape recognition applications including recognizing lines, circles, and ellipses [14]. In [60] an accumulation technique is used to recognize arbitrary shapes by use of the Simple K-Space algorithm (Section 4.6). The subject of this dissertation is primarily an extension and analysis of this method, which is applied to the recognition of sketch-based passwords.

4.5 Shape Matrix and Other Methods

One important representation of shape is Kendall's shape matrix [56], which considers shape to be a property that is invariant to rigid transformations. Kendall's approach to shape first defines a *pre-shape*, which accounts for any translations or changes in scale (or size). Then, shape is considered to be an equivalence class of pre-shapes (represented by the shape matrix), defined by rotations from $\mathbf{SO}(n)$.

A review of multiple shape representations is provided by Zhang and Lu [107]. This review discusses metrics, such as the Hausdorff distance [12, 90], used for comparing shapes. Complex methods using invariant moments [49] and the shape matrix [56] are also described. Lastly, scale space implementations, in particular curvature scale space (CSS) [4, 31] methods, are also used for shape recognition.

4.6 Simple K-Space

Simple K-Space (SKS) is a general approach or philosophy used for matching and recognition applications. The one parameter formulation of SKS [96] is used for matching shapes. There exists a two parameter version of SKS [61], which is used for image registration, however, in

this section only the one parameter method is relevant to the application of shape recognition (and the similar application of sketch-based passwords).

While SKS is an accumulation-based approach, there are some similarities between SKS and Kendall's [56] perspective on shape. Kendall first constructs a translation and scale invariant representation, and then considers a similar shape to be any rotation of that representation. SKS, however, construct a translation and rotation invariant model, and then considers a similar shapes to be a scaled (or zoomed) version of that model.

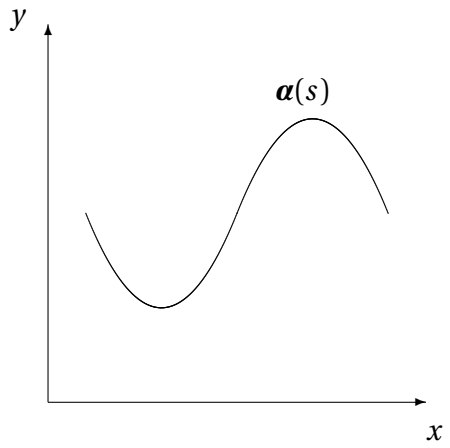


Figure 4.1: A curve in the plane parameterized by arc length.

The single parameter SKS algorithm begins by constructing a model (i.e. a template) of a given shape in the x-y plane (Figure 4.1). A shape is defined as $\boldsymbol{\alpha}(s) = [x(s)\ y(s)]^T$, where the shape is parameterized by a single parameter s—arc length (Equation 4.1). Note that shapes are not parameterized explicitly, however, we provide a parameterization in order to compute features necessary for constructing a model that is independent of the parameterization.

$$s(t) = \int_{t_0}^{t} \|\boldsymbol{\alpha}(u)\| \, \mathrm{d}u \tag{4.1}$$

Using a curve, two new shape functions are computed (or estimated), $\rho_{\boldsymbol{\alpha}}(s)$ and $\kappa_{\boldsymbol{\alpha}}(s)$, where $\rho_{\boldsymbol{\alpha}}(s) = \|\boldsymbol{\alpha}(s) - \mathbf{x}_0\|$ and $\kappa_{\boldsymbol{\alpha}}(s) = |\boldsymbol{\alpha}''(s)|$ is defined as the local curvature at an arc length s. The distance $\rho_{\boldsymbol{\alpha}}(s)$ is with respect to an arbitrary, but constant, spatial reference point, $\mathbf{x}_0$. In practice, $\mathbf{x}_0$ is chosen to be the centroid of the curve $\boldsymbol{\alpha}(s)$. For now, curvature is assumed to be continuously defined over the entire curve, except for points of discontinuity (e.g. endpoints). There are some caveats when using and computing curvature, which are is discussed in Section 6.4 along with details regarding curvature estimation.

Typically, $\rho_{\boldsymbol{\alpha}}(s) \in \mathscr{U}$ and $\kappa_{\boldsymbol{\alpha}}(s) \in \mathscr{K}$, where $\mathscr{U} \subset \mathbb{R}$ defined by $[0,\ \rho_{max}]$ and $\mathscr{K} \subset \mathbb{R}$ defined by $[0,\ \kappa_{max}]$. ρ_{max} and κ_{max} are determined, in practice, by the length of diagonal of the image plane and the resolution (or sampling rate) of the curve (see Chapter 6) respectively.

After computing $\rho_{\boldsymbol{\alpha}}(s)$ and $\kappa_{\boldsymbol{\alpha}}(s)$, the model of a curve, $\boldsymbol{\alpha}(s)$, may be computed using the following function:

$$m_{\boldsymbol{\alpha}}(\rho,\, \kappa) = \frac{1}{z} \int_0^1 \exp\left(-\frac{(\rho - \rho_{\boldsymbol{\alpha}}(s))^2}{2\sigma_\rho^2} \right) \exp\left(-\frac{(\kappa - \kappa_{\boldsymbol{\alpha}}(s))^2}{2\sigma_\kappa^2} \right) \mathrm{d}s \tag{4.2}$$

where z is a normalization factor (see Section 4.8.3), σ_ρ and σ_κ are independent smoothing parameters, and $s \in [0,\, 1]$ without loss of generality. The smoothing parameters, σ_ρ and σ_κ, are usually chosen experimentally. As a rule of thumb, σ_ρ and σ_κ are 5%–20% of ρ_{max} and κ_{max} respectively.

The model function, $m_{\boldsymbol{\alpha}} \colon \mathscr{U} \times \mathscr{K} \to \mathbb{R}$, is a scalar function defined for all ordered pairs $(\rho,\, \kappa)$, and the value $m_{\boldsymbol{\alpha}}(\rho, \kappa)$ represents a *likelihood* that there exists a point on the curve that is a distance of ρ from the reference and has a curvature of κ. If normalized such that $\iint m_{\boldsymbol{\alpha}}(\rho, \kappa)\, \mathrm{d}\rho\, \mathrm{d}\kappa = 1$, then the model may be considered a 2D probability density function.

In fact, Equation 4.2 resembles a kernel density estimator [89]:

$$p(\mathbf{x}) = \frac{1}{h^d} \int_0^1 K\left(\frac{\mathbf{x} - \mathbf{x}(\tau)}{h}\right) \mathrm{d}\tau$$

The goal of constructing a model is to determine the similarity between two curves. The matching process for the SKS algorithm implements an accumulation technique [9] to determine the *consistency* between a shape and model. This consistency measure is used to infer similarity between the shapes.

Given a test curve, $\boldsymbol{\alpha}_t(s) = [x_t(s)\, y_t(s)]^T$ (with curvature function $\kappa_{\boldsymbol{\alpha}_t}(s)$), the *consistency* of $\boldsymbol{\alpha}_t(s)$ with $m_{\boldsymbol{\alpha}}(\rho, \kappa)$ is computed using an accumulator (Equation 4.3).

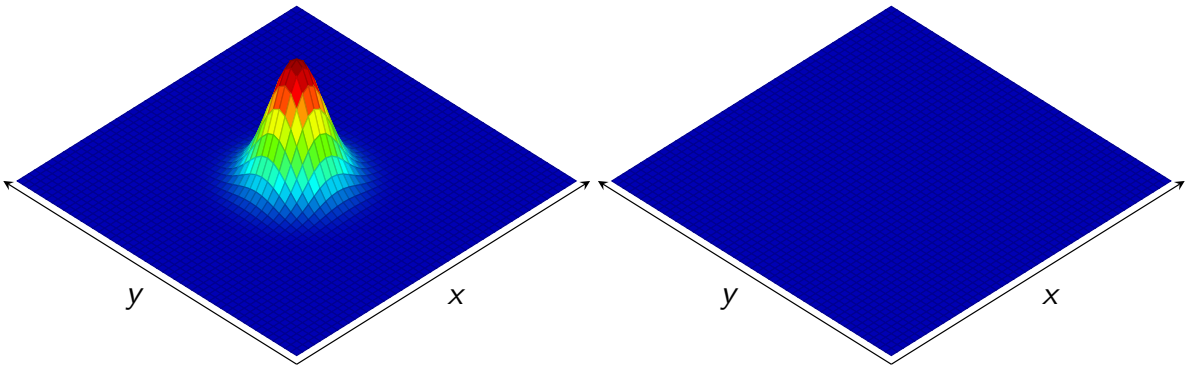
$$\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_t}(\hat{\boldsymbol{x}}) = \int_0^1 m_{\boldsymbol{\alpha}}(\|\hat{\boldsymbol{x}} - \boldsymbol{\alpha}_t(s)\|, \kappa_{\boldsymbol{\alpha}_t}(s))\, \mathrm{d}s \tag{4.3}$$

In principle, the accumulator represents a likelihood that the point $\hat{\boldsymbol{x}} = (\hat{x}, \hat{y})$ is the reference point of a curve, $\boldsymbol{\alpha}_t(s)$, that is consistent with the model. If the curve being compared to the model is the sufficiently similar, then by computing $\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_t}(\hat{\boldsymbol{x}})$ for all $\hat{\boldsymbol{x}}$, a sharp peak will occur at the "best" reference point. Otherwise, the accumulator will not exhibit any significant peaks. Figure 4.2 illustrates the difference between an *ideal* accumulator indicating a matching shape and one indicating no match.

4.7 Generalized Simple K-Space

In this section, the 2D SKS model (Section 4.6) is extended to a model with an arbitrary number of dimensions. In many applications, it may be necessary to model more than simply shape (e.g. direction, order, velocity, acceleration, and etc.).

In the previous section, the model is constructed using two functions: $\rho_{\boldsymbol{\alpha}}(s)$ and $\kappa_{\boldsymbol{\alpha}}(s)$.



(a) Ideal matching accumulator. (b) Ideal non-matching accumulator.

Figure 4.2: If the template and test shapes are similar then the accumulator should exhibit a significant peak (a). Otherwise, no significant peak should occur (b).

Now, consider this to be a vector-valued function (still parameterized by arc length). In the 2D case, the function is $\mathbf{v}_{\alpha}(s) = [\rho_{\alpha}(s)\kappa_{\alpha}(s)]^T$. If the function includes $\rho_{\alpha}(s)$ and n other functions of s (associated with the curve), then the vector-valued function is

$$\mathbf{v}_{\alpha}(s) = \left[\rho_{\alpha}(s) f_{\alpha}^{(1)}(s) \cdots f_{\alpha}^{(n)}(s)\right]^T.$$

From $\mathbf{v}_{\alpha}(s)$ and Equation 4.2, the generalized model becomes

$$m_{\alpha}(\mathbf{v}) = \frac{1}{z}\int_0^1 \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\alpha}(s))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\alpha}(s))\right) \mathrm{d}s \tag{4.4}$$

where

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{\rho}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{f^{(1)}}^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{f^{(n)}}^2 \end{bmatrix}$$

The covariance matrix, $\boldsymbol{\Sigma}$, is diagonal because the features, $\rho_{\boldsymbol{\alpha}}\ f_{\boldsymbol{\alpha}}^{(1)}\ \ldots\ f_{\boldsymbol{\alpha}}^{(n)}$, are all assumed to be independent. In general, feature dependency must be considered. The model, $m_{\boldsymbol{\alpha}} : \mathscr{F} \rightarrow \mathbb{R}$, is defined for all $\mathbf{v} \in \mathscr{F}$, where $\mathscr{F} \subset \mathbb{R}^{n+1}$.

Then, matching is, again, performed by computing the accumulator. Similar to Equation 4.3, the accumulator is

$$\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_t}(\hat{\boldsymbol{x}}) = \int_0^1 m_{\boldsymbol{\alpha}}(\|\hat{\boldsymbol{x}} - \boldsymbol{\alpha}_t(s)\|,\ f_{\boldsymbol{\alpha}_t}^{(1)}(s), \cdots,\ f_{\boldsymbol{\alpha}_t}^{(n)}(s))\ \mathrm{d}s \tag{4.5}$$

In the same way as with the 2D case, a "match" is expected to generate a peak in the accumulator, and a "non-match" produces no significant peaks. This phenomena occurs because the accumulator continuously sums the likelihood measures (from the model) that the example curve has points with the same distances from the reference point and the same properties (e.g. curvature) as the test curve. This process is repeatedly performed for all possible reference points of the test curve. Therefore, assuming the curves are the same, the accumulation attains a maximum at (or near) the actual reference point. On the contrary, if the curves are not similar, then some accumulation may occur. However, the accumulation, in this case, will be relatively small (in comparison).

4.8 Discussion and Analysis

4.8.1 Alternative Model

An alternative to performing a path integral over the curve for computing the model is to use the maximum over the curve. That is,

$$m_{\boldsymbol{\alpha}}(\mathbf{v}) = \frac{1}{z} \max_{s} \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))\right) \tag{4.6}$$

This alternative model is computing the likelihood that at least one point exists on the curve with the set of features $\mathbf{v}$.

Both Equation 4.4 and Equation 4.6 are special cases. If we generalize the right-hand side (RHS) of both equations as

$$\frac{1}{z}\left\{\int_0^1 \left|\exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))\right)\right|^p \mathrm{d}s\right\}^{\frac{1}{p}}$$

then Equation 4.4 is the solution for when $p = 1$, and Equation 4.6 is the solution to the limit as $p \to \infty$.

Experimentally, it has been shown that maximizing (opposed to integrating) performs better in practice [61]. This is probably due to the fact that the integral form in Equation 4.4 includes extraneous information in the model. Despite the fact that the information included tends to be small (on average), the accumulated effects may slightly worsen performance. Whereas, the maximum form (Equation 4.6) incorporates only the "best" matching point into the model. Since the extraneous information is not considered, the accumulator is not affected in the same manner as before. Therefore, this may result in improved performance. This is only conjecture, and should not be considered as a proof.

4.8.2 Invariances

The 2D SKS model works well due to the invariances incorporated into the model. In general, if all features are invariant to translation, rotation, and scale, then the model is invariant to each as well.

Scale invariance is necessary when it comes to shape spaces. Kendall's [55] definition of size (or scale) of a shape is proportional to the average distance from the centroid. Scale invariance is achieved by normalizing the scale to unity. Thus, taking size of the shape or curve out of the equation. This definition of scale is only useful for non-occluded shapes and curves.

Consider the two features used for the 2D model, ρ and κ. ρ, by definition is the distance to a reference point. Invariance to translation and rotation for ρ is achieved if and only if the arbitrary reference point is chosen to both move and rotate with the curve. Similarly, translations and rotations do not effect curvature. Therefore, the model is also invariant to both.

Extending this concept to an arbitrary number of features implies that all features must be invariant in order to ensure that the model is invariant. Each feature must translate and rotate with the curve if the multidimensional model is to be invariant to both translation and rotation. If a feature is not inherently invariant, then it must be adjusted by means of a transformation.

4.8.3 Model Normalization

Normalizing the model, although not a necessity, makes determination of curve similarity much easier. First, consider the accumulator (Equation 4.5). This equation represents a path integral over the model (Figure 4.3).

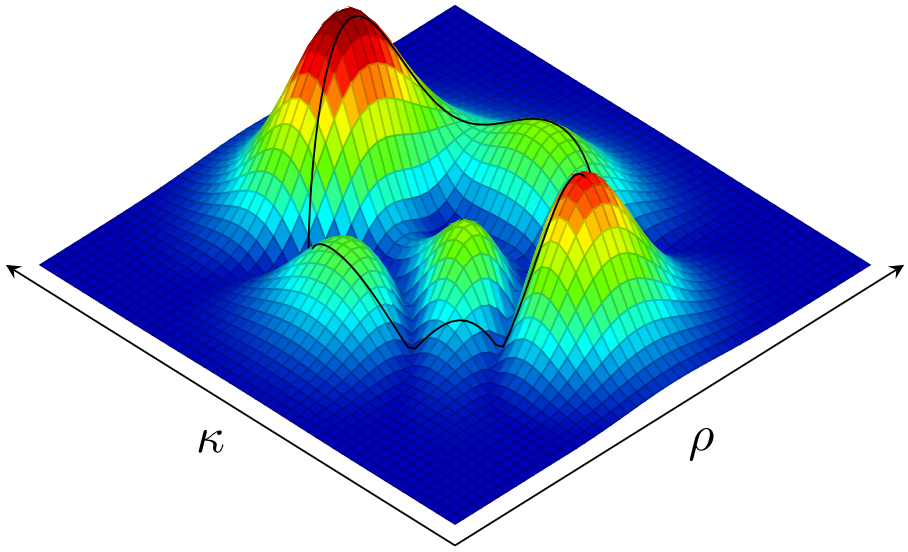


Figure 4.3: The surface represents the model and the line illustrates an integration path, which defined by a curve, over the model. Note that the path is not necessarily closed.

Assuming that the unnormalized model (denoted by $\tilde{m}_{\boldsymbol{\alpha}}$) for $\boldsymbol{\alpha}(s)$ is

$$\tilde{m}_{\boldsymbol{\alpha}}(\mathbf{v}) = \int_0^1 \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))\right) \mathrm{d}s$$

Then, the normalization factor, z, is the path integral over $\tilde{m}_{\boldsymbol{\alpha}}(\mathbf{v})$ or

$$z = \int_0^1 \tilde{m}_{\boldsymbol{\alpha}}\left(\left\|\mathbf{x}^* - \boldsymbol{\alpha}(s)\right\|, f_{\boldsymbol{\alpha}}^{(1)}(s), \cdots, f_{\boldsymbol{\alpha}}^{(n)}(s)\right) \mathrm{d}s \tag{4.7}$$

where the integration path is defined by the curve, $\boldsymbol{\alpha}(s)$, used to construct $\tilde{m}_{\boldsymbol{\alpha}}(\mathbf{v})$.

Normalizing the model $\tilde{m}_{\boldsymbol{\alpha}}$ by the factor z (Equation 4.7) is essentially scaling the accumulator such that the magnitude of $\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}(\mathbf{x}^*) = 1$, where $\mathbf{x}^*$ represents the location of the

peak in the accumulator, when comparing $m_{\boldsymbol{\alpha}}(\mathbf{v})$ with the curve $\boldsymbol{\alpha}(s)$.

On the contrary, if considering $\boldsymbol{\alpha}_t(s)$ to be *different* from $\boldsymbol{\alpha}(s)$ (e.g. non-rigid transformation of $\boldsymbol{\alpha}(s)$), then the magnitude of the peak in the accumulator produced by matching the model, $m_{\boldsymbol{\alpha}}(\mathbf{v})$, with the curve, $\boldsymbol{\alpha}_t(s)$ is less than one. How far below one depends on the severity of the deformation. The more deformed the curve is from the original, the smaller the peak in the accumulator. Therefore, normalizing the model in this manner simplifies the process of determining the similarity between two curves.

4.8.4 Model Complexity

A useful property that exists due to the SKS model definition is that a measure of complexity can be computed from the model. Since the SKS model is essentially a probability distribution (when normalized correctly), entropy—a statistical measure of uncertainty or randomness [24]—can easily be computed. The entropy (in bits), or more precisely the differential entropy, for a continuous random variable, $\mathbf{U}$, is computed as

$$h_2(\mathbf{U}) = -\int_{\mathbb{U}} f(u)\log_2 f(u)\,\mathrm{d}u$$

for a probability distribution function $f(u)$, where $\mathbb{U}$ denotes the set of all possible values for u. Therefore, entropy may be defined over the model in a similar manner.

There is, at least on some level, a relationship between entropy of the SKS model and complexity of the curve used to construct the model. As an illustrative example using a 2D model (with no smoothing, i.e. the Gaussians become delta functions), consider a circle with a radius of ρ and curvature, $\kappa = \frac{1}{\rho}$. The model contains a singular non-zero point at the location (ρ, κ). Therefore, the entropy for an ideal circle is equal to zero. From an information

theoretic perspective, the model has no information; it is deterministic. This also implies that the curve is not very complex. On the other hand, if the model contains two non-zero points (with equal probability)[2], then the entropy is calculated as

$$h_2 = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1$$

Extending the two previous examples to an increasing number of non-zero points in the model, entropy becomes greater than 1. Therefore,

$$\lim_{n\to\infty} h_2 = \lim_{n\to\infty} -\sum_{i=1}^{n}\frac{1}{n}\log_2\frac{1}{n} = \lim_{n\to\infty} -\log_2\frac{1}{n} = \infty$$

shows that as the curve becomes increasingly more complex, the entropy increases toward infinity. Thus, SKS provides a way to statistically measure the complexity of the curve (or model).

Using a set of 10 hand drawn example of four different figures (with varying complexities), the average entropy for each figure is shown in Figure 4.4. Notice that the entropy for a hand drawn circle not equal to that of the perfect circle. This is because it is practically impossible to drawn a perfect circle. Also, notice how a figure 8 is more complex than a circle and signatures are more complex than figure 8s. This figure is a simple example to illustrate that entropy is a measure that correlates with sketch complexity.

[2]This model corresponds to averaging two models corresponding to two circles with different radii.

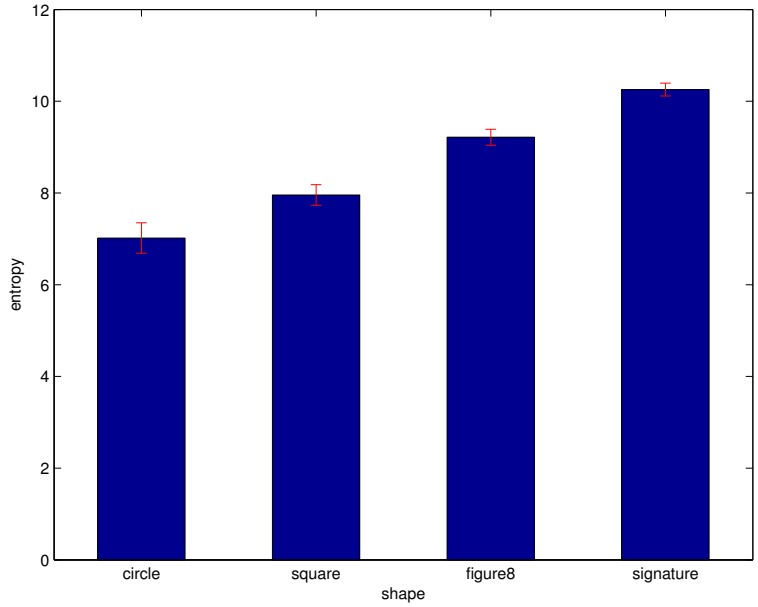


Figure 4.4: The plot shows the average and variation of the entropy for a hand drawn circle, square, figure 8, and signature.

CHAPTER 5

Sketch-based Authentication System

In this chapter, the biometric sketch-based password system is discussed. Much of the material presented in this chapter comes from [87]. The method used for recognizing sketch-based passwords with biometric information in essence works by first constructing a model, and then applying a matching procedure (similar to Equation 4.5) that looks for consistency between a sketch and this model.

In authentication systems, there are two types of problems: *identification* and *verification*. Identification is where a system determines who a user is by using an existing database of users and given sources of information about the user. Verification is where a system determines whether a user is who he/she claims to be by comparing template(s) of the claimed identity with information about the user. Henceforth in this dissertation, the terms recognition and/or matching are implied to mean *verification*.

There are two primary phases which are used in a sketch-based password system (and

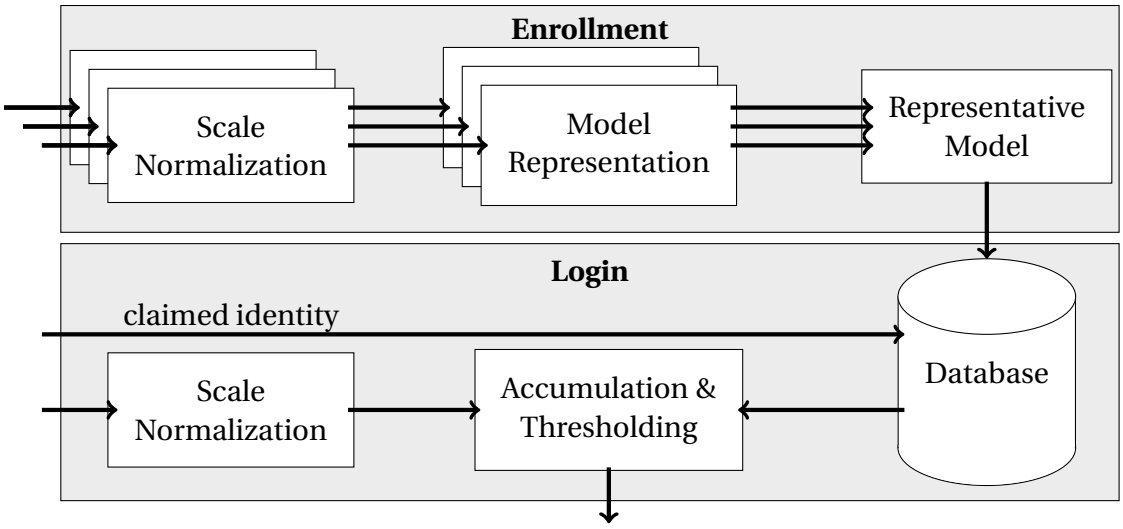


Figure 5.1: The diagram depicts the primary components of both the enrollment and login phases for the proposed sketch-based password system. The inputs (depicted with arrows) to the "Scale Normalization" blocks for both *enrollment* and *login* represent the input sketches from the user for the respective phases.

usually any type of authentication system):

1. Enrollment—User registers using a username and sets his/her sketch-based password.

2. Login—User enters his/her username and draws an instance of the password.

Figure 5.1 outlines the major procedures necessary for both enrollment and login phases. The diagram shows two important steps involved within each phase: capturing the input sketches and normalizing the scale. The next step for enrollment is constructing the model for each normalized sketch, from which a representative model for the class of acceptable sketch-based passwords is produced. This representative model is then stored in the user database.

After normalizing the input sketch during the login phase, the model for the claimed user identity is retrieved from the database. Then, the consistency between the model and the sketch are determined using an accumulative method. If enough evidence exists for the

sketch to be considered similar to the exemplar sketches that constructed the model, then access is granted. Otherwise, access is denied.

5.1 Input and Normalization

The input of the sketch-based password system is simply a sketch. More rigorously, a parameterized sketch (excluding biometrics for now) is considered to be a continuous mapping $\boldsymbol{\alpha}: I \rightarrow \mathbb{R}^2$, expressed as $\boldsymbol{\alpha}(s) = [x(s), y(s)]^T$ for arc length $s \in I$, where I is the interval $(0,1)$ without loss of generality.

Given a sketch, the scale of a sketch $\boldsymbol{\alpha}(s)$ is defined in Equation 5.1, which is similar to the definition used in Chapter 4. Note that without loss of generality the sketch $\boldsymbol{\alpha}$ is assumed to be centered[3] at the location $(0,0)$.

$$\|\boldsymbol{\alpha}\| = \int_0^1 \|\boldsymbol{\alpha}(s)\| \, \mathrm{d}s \tag{5.1}$$

In order to achieve scale invariance, every sketch is normalized to unity scale. Given an unnormalized, but centered, sketch, $\boldsymbol{\alpha}^*(s)$, the sketch is normalized by

$$\boldsymbol{\alpha}(s) = \frac{\boldsymbol{\alpha}^*(s)}{\|\boldsymbol{\alpha}^*\|}$$

Therefore, normalization results in $\|\boldsymbol{\alpha}\| = 1$.

The definition of scale (Equation 5.1), which is similar to Kendall's definition [55], is useful only for non-occluded sketches. In the application of sketch-based passwords, occluded sketches are considered to be different or incomplete passwords. Therefore, occlusion is not considered in this application.

[3]The sketch is centered by translating the coordinate axes to the the centroid of $\boldsymbol{\alpha}$.

5.2 Enrollment

During enrollment, the user is expected to provide multiple instances of a sketch-based password. Multiple sketches are required in order to construct a more accurate and robust model than a model produced by a single sketch. The main idea is to capture more of the possible variations from a particular user, which helps to produce a model that is consistent with sketch-based passwords generated by that particular user. This is, in fact, similar to traditional text-based schemes. In most text-based password systems, users are required to enter his/her desired password at least twice. Therefore, it is not unreasonable for a sketch-based password system to demand multiple sketches during enrollment. Due to the additional complexity of a sketch compared with an alphanumeric string, it is expected that a sketch would require more than two examples for enrollment (e.g. 3–5).

Given a normalized sketch, the goal is to construct a model or descriptor that captures the desired properties of a sketch, which include:

- shape (e.g. relative distance and curvature)
- drawing direction (e.g. sketch tangent direction)
- biometrics (e.g. pressure, velocity, or acceleration)
- parametrization (e.g. time or arc length).

The model we have chosen is a biometric extension and generalization of SKS [96, 61]. A generalization of SKS is implemented because it has been shown to be robust to the multitude of variations to shape contours [60]. Since a sketch may, for all intents and purposes, be considered a shape, an extension to SKS which models shape and other properties seems appropriate.

5.2.1 Features

Before discussing the model construction, several local features of the sketch are required. These features are intended to capture the local properties discussed above.

Two shape functions, $\rho_{\boldsymbol{\alpha}}(s)$ and $\kappa_{\boldsymbol{\alpha}}(s)$, are computed (or estimated) from the normalized sketch, $\boldsymbol{\alpha}(s)$. $\rho_{\boldsymbol{\alpha}}(s) = \|\boldsymbol{\alpha}(s)\|$ is defined as the Euclidean distance between the spatial coordinates at arc length s along the sketch, $\boldsymbol{\alpha}(s)$, and an arbitrary, but constant spatial reference point[4]. In this approach, we have chosen the reference point of the sketch to be the center of gravity. Then, $\kappa_{\boldsymbol{\alpha}}(s)$ is defined as the local curvature at the point s on the sketch. Curvature is defined as the magnitude of the derivative to the unit tangent vector, or $\kappa(s) = |T'(s)| = |\boldsymbol{\alpha}''(s)|$.

Typically, $\rho_{\boldsymbol{\alpha}}(s) \in \mathscr{U}$ and $\kappa_{\boldsymbol{\alpha}}(s) \in \mathscr{K}$, where $\mathscr{U} \subset \mathbb{R}$ defined by $[0,\ \rho_{max}]$ and $\mathscr{K} \subset \mathbb{R}$ defined by $[0, \kappa_{max}]$. In practice, ρ_{max} and κ_{max} are determined respectively by the length of the diagonal of the active drawing area (x-y plane) and the device resolution.

Note that both features, distance and curvature, are invariant to translation, rotation, and reflections. Also, when normalized for size a priori they are also scale invariant. This will be important in Section 5.4.2.

In order to capture the drawing direction at each point along the sketch, $\boldsymbol{\alpha}(s)$, the tangent direction to the sketch is computed. The tangent direction at a point s along the sketch is represented by a scalar angle $\theta_{\boldsymbol{\alpha}}(s) \in \Theta$, where $\theta_{\boldsymbol{\alpha}}(s) = \text{atan2}(y'(s), x'(s))$ and Θ is defined by the interval $(-\pi, \pi]$.

Since sketches drawn by most users are very noisy (in terms of jitter caused by shaking of the hand), $\theta_{\boldsymbol{\alpha}}(s)$ is significantly impacted by the amount of jitter. There are two ways to handle jitter in the drawing direction: 1) quantizing the direction angle, and 2) smoothing the jitter using a convolution kernel. In our approach, we use the latter because noise

[4]Without loss of generality, the sketch is assumed to be centered at the origin. Therefore, the reference point is omitted.

near a quantization level boundary introduces significant errors. The direction feature is not smoothed directly, however, SKS includes a parameter that blurs all features (not just direction) in order to be more robust.

The next property of a sketch that is modeled is the biometric component—pressure[5]—which is denoted by the function $b_{\boldsymbol{\alpha}}(s)$. The pressure, $b_{\boldsymbol{\alpha}}(s) \in \mathscr{B}$ (where $\mathscr{B}$ is defined as the interval $[0,\ 1]$), is a scalar measurement representing the amount of force applied by the pen with the active area on the device's screen.

Finally, the arc length is also used as a feature. Here, arc length is defined in the same way as Equation 4.1. The arc length is used primarily to impose a general ordering of the sample of the sketch (i.e. how a sketch drawn). However, it also significantly improves performance. The arc length without loss of generality is on the interval $\mathscr{S}$, defined by $[0,\ 1]$.

5.2.2 Model

After computing each of the functions: $\rho_{\boldsymbol{\alpha}}(s)$, $\kappa_{\boldsymbol{\alpha}}(s)$, $\theta_{\boldsymbol{\alpha}}(s)$, $b_{\boldsymbol{\alpha}}(s)$, and $s_{\boldsymbol{\alpha}}(t)$, the model of a sketch, $\boldsymbol{\alpha}(s)$, (similar to Equation 4.4) is defined as:

$$m_{\boldsymbol{\alpha}}(\mathbf{v}) = \frac{1}{z} \int_0^1 \exp\left(-\frac{1}{2} (\mathbf{v} - \mathbf{v}_{\boldsymbol{\alpha}}(\tau))^T \boldsymbol{\Sigma}^{-1} (\mathbf{v} - \mathbf{v}_{\boldsymbol{\alpha}}(\tau)) \right) \mathrm{d}\tau \tag{5.2}$$

where $\mathbf{v}_{\boldsymbol{\alpha}}(\tau) = \left[\rho_{\boldsymbol{\alpha}}(\tau)\ \kappa_{\boldsymbol{\alpha}}(\tau)\ \theta_{\boldsymbol{\alpha}}(\tau)\ b_{\boldsymbol{\alpha}}(\tau)\ s_{\boldsymbol{\alpha}}(\tau) \right]^T$ is the local feature vector at the point τ on the sketch (and τ is just a "dummy" arc length variable), z is a normalization factor (see Section 5.3.1), and $\boldsymbol{\Sigma}$ is the covariance matrix used for smoothing the model. The intent behind blurring the model is to make the matching of a sketch during the login phase (Section 5.3) more robust. This is discussed further in Section 5.4.3.

[5] In order, to use pressure as a feature, the digitizer or tablet must have either a capacitive touchscreen or additional hardware for estimating or measuring the pressure level.

In this work, $\mathbf{\Sigma}$ is assumed to be a diagonal covariance matrix, where the diagonal elements are user defined parameters for smoothing each feature independently. The non-zero parameters in the matrix are determined experimentally. The diagonal covariance matrix assumes that the features used are independent. In general, feature dependency must be considered, but whether or not the covariance matrix is diagonal the purpose remains to smooth the model. In Section 5.4, we show how the covariance matrix is used for controlling the security and usability of the system.

The model in Equation 5.2, $m_{\boldsymbol{\alpha}}\colon \mathscr{F} \to \mathbb{R}$, is a scalar function defined for all vectors $\mathbf{v} = \begin{bmatrix}\rho & \kappa & \theta & b & s\end{bmatrix}^T \in \mathscr{F}$, where $\mathscr{F} \subset \mathscr{U} \times \mathscr{K} \times \Theta \times \mathscr{B} \times \mathscr{S}$ denotes the "feature space." The scalar value represents a *likelihood* that there exists a point, at an arc length of s on $\boldsymbol{\alpha}$, that is a distance of ρ from the reference, has a curvature of κ, has a drawing direction of θ, and has a pressure of b.

The model, in fact, may be viewed[6] as hyper-surface defined over $\mathscr{F} \subset \mathbb{R}^5$, and it very much resembles a kernel density estimator [89]. However, the main difference is the normalization factor, z. Density estimators normalize such that

$$\int_{\mathscr{F}} m_{\boldsymbol{\alpha}}(\mathbf{v})\, \mathrm{d}\mathbf{v} = 1,$$

instead, an alternative method of normalization, which is better suited to this application, is used.

5.2.3 Representative Model

After constructing each model from the multiple sketches drawn by the user during the enrollment stage, it is useful to construct a single representative model (opposed to matching

[6]Alternatively, it may be thought of as every point along the sketch having a hyper-surface defined over $\mathbb{R}^4$

with all exemplars). This is accomplished by averaging the set of models (Figure 5.2).

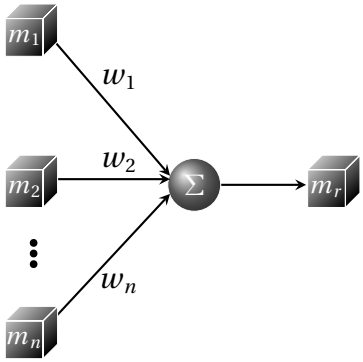


Figure 5.2: Computing the representative model for a set of input sketches. The nodes on the left represent the individual models constructed from the input sketches and the output node symbolizes the representative model.

Manifold Assumption

At first this approach may appear to be trivial, however, the reason behind it is not intuitive. Here, the assumption is made that the "space of sketches" has the properties of a manifold:

- locally Euclidean
- locally "smooth"

The assumption is that sketches input by the user during enrollment must be *similar* sketches. This means the sketches must be of the same shape and drawn in a similar manner. Therefore, on the manifold of sketches, they are considered to be "close" in terms of their geodesic distance. Since the models are constructed from these sketches, they too lie in "close" proximity on the manifold of models. The average is then an adequate representative model for the class because manifolds are locally linear and the average optimally minimizes

the distance from each point in linear space. However, if a user enters extremely different sketches during enrollment then the average model is a poor representative of the set of sketches. Therefore, a simple outlier test to determine if the sketches provided by the user are sufficiently different is implemented.

Outlier Detection

According to [72], in general, an outlier is an observation that lies outside the overall pattern of a distribution. An *outlier* in the context of sketch-based passwords is defined as: an instance of a sketch that deviates from (or does not conform to) the statistically significant pattern which is exhibited from the distribution of sketches.

The most prevalent outlier detection method is RANdom SAmple Consensus (RANSAC) [34], which is an iterative method that is able to effectively estimate parameters of some model given sample data containing outliers. The RANSAC algorithm is summarized in Algorithm 1.

Given a set of observations (e.g. a set of sketches from an individual), D, the first step of the RANSAC algorithm is to randomly select a subset n observations (labeled Q) for computing the parameters of some model. Note that n must be greater than or equal to the minimum number of observations required for estimating the model parameters. Second, RANSAC uses the subset Q for computing the model parameters. Then, using randomly constructed model and the input set D, the inliers I are determined as the observations that are within some user defined margin of error ε of the model, which is usually experimentally chosen. If the number of inliers (denoted by $\#(I)$) is greater than a experimentally chosen threshold ζ and if also greater than the maximum number of inliers observed, then the maximum number of inliers is updated and the model parameters are saved. Lastly, the process is repeated a predetermined number of times.

Algorithm 1 RANSAC

Require: a set of input observations D, which possibly include outliers

```
I_max ← 0
P* ← { }
while iter < N do
    Q ← n_random_observations(D)
    P ← Compute_Params(Q)
    I ← Find_Inliers(P,D)
    if #(I) > ζ then
        if #(I) > I_max then
            I_max ← #(I)
            P* ← Compute_Params(I)
        end if
    end if
end while
return P*
```

RANSAC is only a general philosophy for finding a consistent set of observations. Note that it does not explicitly define what constitutes as consistent; that is left up to the designer. Therefore, the model type and algorithm thresholds must be chosen by the one implementing RANSAC, based on the specific application. For more information on RANSAC see [34].

Although RANSAC can easily be adapted to detect outliers from the set of training sketches, a simpler approach is used. At the core of the RANSAC algorithm inliers and outliers are determined by some margin of error, according to some definition of error. Since the number of training sketches is intentionally limited to 3–5 exemplars, it is easier to threshold the similarity of each sketch with the representative model of the set (Figure 5.3). Note that this similarity measure comes from the SKS algorithm which has already been discussed in detail.

An interesting observation is that the optimal sketch necessary to produce to the representative model is unknown to the user after enrollment, unless the user is perfectly consistent (practically impossible). However, there is an interesting the result that occurs (Observation 1)

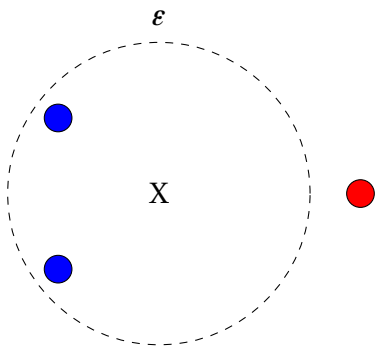


Figure 5.3: The blue dots represent sketches that are considered inliers,the red dot represents an outlier, the X represents the representative for the set. The inliers are determined by the predetermined margin of error, ε.

Observation 1. *Users are able to learn what the system expects as a "correct" sketch.*

Similar to text-based systems, users have a better chance of being granted access the more they use a password. For sketch-based password, Observation 1 also follows because as users frequently use the system, they receive some feedback (e.g. visual and muscle memory) for what constitutes as a sufficiently similar sketch. Over time, the feedback enable the users to login using the sketch with little to no effort. Although this not the primary focus of this work, it is an interesting process that occurs during the human-computer interaction.

Now that the enrollment phase has been discussed, we move on to discuss the login phase which is the primary mode of operation.

5.3 Login

During the login phase, a user inputs his/her putative identity using a username or pin, and then the user draws the sketch-based password (attempting to replicate the underlying biometric signature used during enrollment). This sketch is then determined to be either sufficiently or insufficiently consistent with the representative model of the putative user.

The method used for computing this measure of consistency is the higher dimensional extension to the matching procedure used by SKS (Section 4.7). The SKS approach uses an accumulative framework for determining the consistency of a shape with a shape model. Thus, the approach used here is very similar for sketches.

Given a login sketch, $\boldsymbol{\alpha}_\ell(s) = \left[x_\ell(s),\, y_\ell(s)\right]^T$ (with corresponding curvature, direction, pressure, and arc length functions) and model $m_{\boldsymbol{\alpha}}(\mathbf{v})$, the accumulator is defined as

$$\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}}) = \int_0^1 m_{\boldsymbol{\alpha}}(\mathbf{v}_{\boldsymbol{\alpha}_\ell}(\tau, \hat{\mathbf{x}}))\, \mathrm{d}\tau \tag{5.3}$$

where

$$\mathbf{v}_{\boldsymbol{\alpha}_\ell}(\tau, \hat{\mathbf{x}}) = \left[\|\hat{\mathbf{x}} - \boldsymbol{\alpha}_\ell(\tau)\|\ \kappa_{\boldsymbol{\alpha}_\ell}(\tau)\ \theta_{\boldsymbol{\alpha}_\ell}(\tau)\ b_{\boldsymbol{\alpha}_\ell}(\tau)\ \tau_{\boldsymbol{\alpha}_\ell}(\tau)\right]^T$$

is the feature vector defined by $\boldsymbol{\alpha}_\ell$ and $\hat{\mathbf{x}}$; $\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}})$ represents the likelihood that $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$ is the reference point of the sketch, $\boldsymbol{\alpha}_\ell$. This likelihood may be thought of as being the result from a path integral over the hyper-surface of the model, $m_{\boldsymbol{\alpha}}$, where the path is defined by $\hat{\mathbf{x}}$ and the local features of the login sketch. An example of this path integration using a 2D model (higher dimensional case is more difficult to show) is shown in Figure 5.4.

Assuming that $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_\ell$ are, indeed, two similar sketches, then the consistency measure increases as $\hat{\mathbf{x}}$ gets closer to the the actual reference point of $\boldsymbol{\alpha}_\ell(s)$. Thus, producing a global maximum in the accumulator at the location of the "best" reference point. On the other hand, if they are sufficiently different then no significant peak[7] will occur. Figure 5.5 shows the difference between matching and non-matching accumulators.

Since the amplitude at every point in the accumulator represents a likelihood measure of that point being the reference point, then the best reference point occurs at $\mathbf{x}^* = \arg\max_{\mathbf{x}} A(\mathbf{x})$.

[7]The accumulator may have other local maxima, but typically their amplitudes are small in comparison to that of a peak generated from a similar sketch.

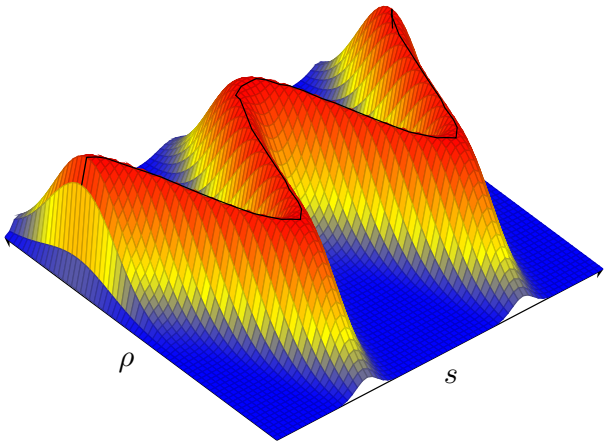


Figure 5.4: The surface represents the model constructed over the ρ-s feature space for an ellipse, and the line represents an integration path defined by $\hat{\mathbf{x}}$ and the features of $\boldsymbol{\alpha}_\ell$. The result from each integral path represents the likelihood that $\hat{\mathbf{x}}$ is the reference point for the model sketch.

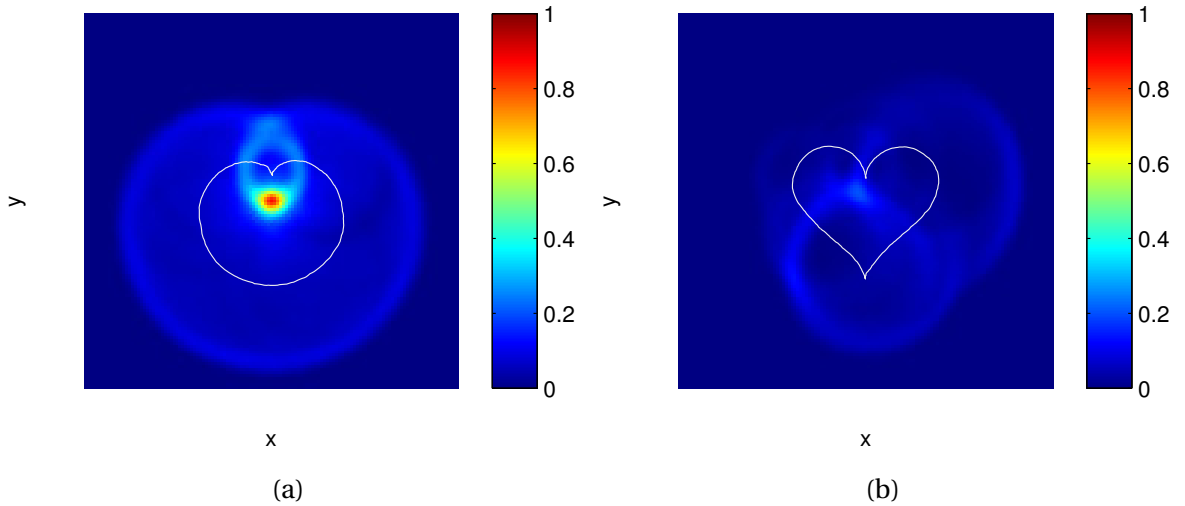
Thus, the authentication decision is made as follows:

$$decision = \begin{cases} \text{ACCESS GRANTED}, & A(\mathbf{x}^*) > Thresh \\ \text{ACCESS DENIED}, & A(\mathbf{x}^*) \leq Thresh \end{cases}$$

which means that if the login sketch, $\boldsymbol{\alpha}_\ell$, is sufficiently consistent (the accumulator peak is greater than than the user-defined and experimentally determined threshold, *Thresh*) with the model for $\boldsymbol{\alpha}$, then the two sketches are considered to be similar.

5.3.1 Model Normalization

Normalizing the model, although not a necessity, makes determination of sketch consistency with a model much easier. Consider the unnormalized model (i.e. where $z = 1$) for the sketch



(a) (b)

Figure 5.5: Example of an accumulator that indicates a sketch (shown on top of the accumulator) that is consistent with the model (a), and an accumulator that indicates a different sketch (also shown) is inconsistent with the same model (b).

$\boldsymbol{\alpha}(s)$:

$$\tilde{m}_{\boldsymbol{\alpha}}(\mathbf{v}) = \int_0^1 \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(s))\right) \mathrm{d}s.$$

The most consistent sketch with $\tilde{m}_{\boldsymbol{\alpha}}$ is $\boldsymbol{\alpha}(s)$. Therefore, the normalization factor

$$z = \int_0^1 \tilde{m}_{\boldsymbol{\alpha}}(\mathbf{v}_{\boldsymbol{\alpha}}(s,\mathbf{x}^*))\,\mathrm{d}s,$$

which represents result of integrating on the path (defined by the best reference point $\mathbf{x}^*$) over the unnormalized model. Using this definition of z, the model is normalized such that $\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}(\mathbf{x}^*) = 1$ (i.e. the consistency between the model of $\boldsymbol{\alpha}(s)$ and $\boldsymbol{\alpha}(s)$ itself is maximal). In practice, this means if a sketch is consistent with the model, then the magnitude of the peak in the accumulator should be close to one. Otherwise, it should be bounded away from one. How far below one depends on the severity of the deformation from the original sketch. The more deformed the sketch, the smaller the peak in the accumulator. Therefore, normalizing

the model in this manner simplifies the process of determining the consistency of the sketch.

5.4 Efficiency, Security, and Usability

In this section, various properties and theorems are discussed, including the following claims: duality and computational complexity of constructing a model, uniqueness of a model, and tolerance in the accumulator. Every claim introduced here theoretically demonstrates that the sketch-based password system is efficient, secure, and robust.

5.4.1 Duality and Computational Complexity

One important property of SKS is the duality for building a model of a sketch. The direct implementation as expressed in Equation 5.2 is called the primal form, and it is implemented in Algorithm 2.

Algorithm 2 Primal

$m_{\boldsymbol{a}}(\mathbf{v}) \leftarrow 0 \quad \forall \mathbf{v}$ ▷ Initialize
for $\mathbf{v} \in \mathscr{F}$ **do**
  for $\tau \in [0, 1]$ **do**
    $m_{\boldsymbol{a}}(\mathbf{v}) \leftarrow m_{\boldsymbol{a}}(\mathbf{v}) + \exp\left(-\frac{1}{2}(\mathbf{v} - \mathbf{v}_{\boldsymbol{a}}(\tau))^T \boldsymbol{\Sigma}^{-1} (\mathbf{v} - \mathbf{v}_{\boldsymbol{a}}(\tau))\right)$
  end for
end for

There also exists a dual to the primal algorithm, in which the sketch is only iterated over once rather than multiple times (as with the primal form). The dual is outlined in Algorithm 3

Both primal and dual algorithms implement the same integrals, but the manner in which the integrals are computed distinguishes them from one another. The primal form computes

Algorithm 3 Dual

$m_{\boldsymbol{\alpha}}(\mathbf{v}) \leftarrow 0 \quad \forall \mathbf{v}$ ▷ Initialize
for $\tau \in [0, 1]$ **do**
  for $\mathbf{v} \in \mathscr{F}$ **do**
    $m_{\boldsymbol{\alpha}}(\mathbf{v}) \leftarrow m_{\boldsymbol{\alpha}}(\mathbf{v}) + \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(\tau))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(\tau))\right)$
  end for
end for

the model by integrating over the sketch for every point, $\mathbf{v}$, in the model. However, the dual iterates only over the model for every point along the sketch. Note the subtle distinction and that both primal and dual have exactly the same algorithmic complexity. However, the dual can be used to construct an approximate model more efficiently, as we explain below.

The dual form (Algorithm 3) provides a way to efficiently construct a good approximation of the model. For each point on the sketch, every point $\mathbf{v}$ is updated in the model by a factor:

$$\exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(\tau))^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(\tau))\right).$$

This factor represents a Gaussian centered at some location in the model, which is defined by the local features on the sketch, denoted by $\mathbf{v}_{\boldsymbol{\alpha}}(\tau)$. Therefore, only points in the model that are within a reasonable local neighborhood of the center of this Gaussian need updating, which significantly reduces the computational complexity for computing the model. Using a finite region of support for the Gaussian, we can show that the dual algorithm scales linearly with the number of points along the sketch (Observation 2). For continuous sketch, the number of points are technically infinite, but in practice the sketches are sampled to a finite number of samples (Chapter 6).

Observation 2. *The complexity for constructing the model is $\mathcal{O}(p)$, where p is the number of samples along the sketch.*

Proof. For each of the p points along the sketch, every point inside the local neighborhood of the corresponding features of p is updated. Thus, the computational complexity is $\mathcal{O}(p\mathcal{N}_p)$ where $\mathcal{N}_p$ denotes the size of the neighborhood or region of support. Assuming that the neighborhood size, defined by the set of points inside the $2\boldsymbol{\Sigma}$ or $3\boldsymbol{\Sigma}$ hyper-ellipsoid, is fixed for all p, then $\mathcal{N}_p$ is constant and the complexity scales linearly with the number of samples over the sketch. Therefore, the complexity reduces to $\mathcal{O}(p)$. □

5.4.2 Model Uniqueness in Limit

An interesting result of using the SKS model (Equation 5.2) is the property of having a unique model (in the limit). This comes primarily from the intuition behind the shape recognition model, but it is also applicable to the generalization discussed in this section.

Consider the limit of the SKS model of a parameterized curve as the elements of the non-negative definite matrix $\boldsymbol{\Sigma}$ approach 0 (i.e. $\boldsymbol{\Sigma} \to \mathbf{0}$ where $\mathbf{0}$ is the matrix with all zero elements)[8], which means there is no smoothing of the model:

$$\lim_{\boldsymbol{\Sigma}\to\mathbf{0}} m_{\boldsymbol{\alpha}}(\mathbf{v}) = \frac{1}{z}\int_0^1 \delta\left(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}}(\tau)\right)\,\mathrm{d}\tau$$

where

$$\delta(\mathbf{x}) = \lim_{\boldsymbol{\Sigma}\to\mathbf{0}} \exp\left(-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}\right) = \begin{cases} 1, & \mathbf{x}=\mathbf{0} \\ 0, & \mathbf{x}\neq\mathbf{0} \end{cases}.$$

Now, consider a mapping $\psi : \mathscr{P} \to \mathscr{M}$ (defined by the model), which maps an element of $\mathscr{P}$—the "space of sketches"—to the "space of models," $\mathscr{M}$. Note the distinction between $m_{\boldsymbol{\alpha}}(\mathbf{v})$, which maps an element of $\mathscr{F}$ to $\mathbb{R}$, and ψ, which maps a "sketch" to some hyper-surface defined by $(\mathbf{v}, m_{\boldsymbol{\alpha}}(\mathbf{v}))$ in $\mathscr{M}$.

[8] Assume that $\lim_{\boldsymbol{\Sigma}\to\mathbf{0}} \boldsymbol{\Sigma}$ is equivalent to $\lim_{\alpha\to 0} \alpha\boldsymbol{\Sigma}$.

Here, we show that the mapping ψ is a one-to-one mapping in the limit, which implies the smoothing matrix $\mathbf{\Sigma}$ may be used to set the trade-off between a unique model or tolerant model (i.e. security vs. usability).

Theorem 1. *In the limit as* $\mathbf{\Sigma} \rightarrow \mathbf{0}$*, two models* $\psi(\boldsymbol{\alpha}_1)$ *and* $\psi(\boldsymbol{\alpha}_2)$ *of two sketches,* $\boldsymbol{\alpha}_1$ *and* $\boldsymbol{\alpha}_2$ *respectively, are equal iff* $\boldsymbol{\alpha}_2 = T_{iso}(\boldsymbol{\alpha}_1)$*, where* T_{iso} *is some isometry (e.g. translation, rotation, or reflection).*

In order to prove Theorem 1, the following lemma is necessary.

Lemma 1. *Given that features composing the feature vector* $\mathbf{v}(s)$ *are invariant to any isometry, then* $\mathbf{v}_1(s) = \mathbf{v}_2(s)$ *iff* $\boldsymbol{\alpha}_2 = T_{iso}(\boldsymbol{\alpha}_1)$.

Proof. First, we need to make sure that the assumption about the feature vectors holds. In the case for sketches, ρ which is a distance from the center of gravity is invariant to translation, rotation, and reflection; as is the curvature κ. The tangent direction θ is not directly invariant to rotation or reflection. However, the angle can be measured relative to the principle direction (which rotates and reflects with the sketch). Therefore, θ may also be invariant to isometries. The biometric b is also invariant to isometries because it translates, rotates, and reflects with the sketch, and arc length is also invariant. Therefore, with sketches we may ensure that the features are invariant to any isometry.

Assuming that $\boldsymbol{\alpha}_2 = T_{iso}(\boldsymbol{\alpha}_1)$, then due to the invariance to isometries $\mathbf{v}_1(s) = \mathbf{v}_2(s)$. And if $\mathbf{v}_1(s) = \mathbf{v}_2(s)$, then $\boldsymbol{\alpha}_2$ is equal to $\boldsymbol{\alpha}_1$ (up to a rigid transformation). □

Now, we can prove Theorem 1.

Proof. Assume that $\boldsymbol{\alpha}_2 = T_{iso}(\boldsymbol{\alpha}_1)$. From Lemma 1, this implies that $\mathbf{v}_1(s) = \mathbf{v}_2(s)$. Therefore,

$$\begin{aligned}\lim_{\boldsymbol{\Sigma}\to\mathbf{0}} m_{\boldsymbol{\alpha}_1}(\mathbf{v}) &= \frac{1}{z}\int_0^1 \delta\left(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}_1}(\tau)\right)\,\mathrm{d}\tau \\ &= \frac{1}{z}\int_0^1 \delta\left(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}_2}(\tau)\right)\,\mathrm{d}\tau \\ &= \lim_{\boldsymbol{\Sigma}\to\mathbf{0}} m_{\boldsymbol{\alpha}_2}(\mathbf{v}).\end{aligned}$$

Thus, $\psi(\boldsymbol{\alpha_1}) = \psi(\boldsymbol{\alpha_2})$.

Conversely, we want to show that $\psi(\boldsymbol{\alpha}_1) = \psi(\boldsymbol{\alpha}_2)$ implies $\boldsymbol{\alpha}_2 = T_{iso}(\boldsymbol{\alpha}_1)$. However, if the contrapositive:

$$\boldsymbol{\alpha}_2 \neq T_{iso}(\boldsymbol{\alpha}_1) \Rightarrow \psi(\boldsymbol{\alpha}_1) \neq \psi(\boldsymbol{\alpha}_2)$$

is true, then the original statement itself is also true.

So, assuming that $\boldsymbol{\alpha}_2 \neq T_{iso}(\boldsymbol{\alpha}_1)$, then we know that $\mathbf{v}_1(s) \neq \mathbf{v}_2(s)$ from Lemma 1. Therefore, $\exists s^*$ s.t. $\mathbf{v}^* = \mathbf{v}_1(s^*) \neq \mathbf{v}_2(s^*)$. Then, it follows that:

$$\begin{aligned}\lim_{\boldsymbol{\Sigma}\to\mathbf{0}} m_{\boldsymbol{\alpha}_1}(\mathbf{v}^*) &= \frac{1}{z}\int_0^1 \delta\left(\mathbf{v}^*-\mathbf{v}_{\boldsymbol{\alpha}_1}(\tau)\right)\,\mathrm{d}\tau \\ &= \frac{1}{z}\delta\left(\mathbf{v}^*-\mathbf{v}_{\boldsymbol{\alpha}_1}(s^*)\right) = \frac{1}{z}\end{aligned}$$

and

$$\begin{aligned}\lim_{\boldsymbol{\Sigma}\to\mathbf{0}} m_{\boldsymbol{\alpha}_2}(\mathbf{v}^*) &= \frac{1}{z}\int_0^1 \delta\left(\mathbf{v}^*-\mathbf{v}_{\boldsymbol{\alpha}_2}(\tau)\right)\,\mathrm{d}\tau \\ &= \frac{1}{z}\delta\left(\mathbf{v}^*-\mathbf{v}_{\boldsymbol{\alpha}_2}(s^*)\right) = 0\end{aligned}$$

Thus, $\psi(\boldsymbol{\alpha}_1) \neq \psi(\boldsymbol{\alpha}_2)$. □

This concept of uniqueness does not theoretically hold for arbitrary $\mathbf{\Sigma}$. However, the system is designed intentionally so that there is some level of tolerance—*non-uniqueness*—in the model. This is due to the fact that people are not perfect; they make errors when reproducing a sketch. So, in order to make the system more robust and usable, the model is intentionally fuzzy and not unique. In fact, true uniqueness is not desirable.

Intuitively, we are claiming that, in general, uniqueness is only a local property of the model. We provide some experimental evidence to support this claim in Section 5.5.1.

5.4.3 Robustness in the Accumulator

Given a model $m_{\boldsymbol{\alpha}}(\mathbf{v})$, there exists some continuous path $\mathbf{p}$ (Figure 5.4) over the model, that is the $\mathbf{p}$ is defined in terms of the features, that maximizes the peak in the accumulator (Equation 5.3). In principle, this path is defined by $\boldsymbol{\alpha}$. Therefore, there exists a small "band" or "tube" around the path $\mathbf{p}$ that defines allowable deviation from the optimal path (corresponding to deviations of the sketch) over the model, which results in a "small" difference in the accumulator peak.

Primarily due to jitter and other noise factors while drawing, most people cannot perfectly reproduce their own sketch. A significant (but less than optimal) amount of accumulation occurs when a sketch, corresponding to a near optimal path over the model, is drawn. Therefore, there is some amount of tolerance (i.e. robustness) built directly into the matching procedure. This tolerance, which is also controlled by $\mathbf{\Sigma}$, is what makes this system more practical. Thus, more blurring in the model implies less security of the system because less accuracy is required of any user (genuine or forger).

5.5 Experiments and Results

In this section, we demonstrate the feasibility and security of our system through experimentation. In order to demonstrate the current level of performance, several experiments are performed using both synthetic and hand-drawn sketches. The purpose of these experiments is two-fold: 1) to support the uniqueness and robustness claims in Section 5.4 and 2) to provide a performance analysis of the proposed password system.

In this dissertation, as well as many other biometric systems, performance is measured by the false acceptance rate (FAR) versus the false rejection rate (FRR) curve. The FAR is the number of accepted forgeries divided by the total number of forgeries, and the FRR is the number of genuine sketches that are rejected divided by the total number of genuine sketches. In some cases, performance is reported with a single number referred to as the equal error rate (EER), which is the rate where the FAR and FRR are equal.

5.5.1 Model "Uniqueness"

In this set of experiments, we demonstrate that the SKS model is, for all intents and purposes, unique. The idea is to show that a "small" change in the sketch results in a "small" change in the model. These experiments provide some experimental evidence to support Theorem 1.

Given two *distinct*—drawn by different individuals—sketches, we compute 50 intermediate sketches using linear interpolation. This allows us to consider 50 unique sketches and their corresponding models from one sketch to another. The measures used to quantify the differences between the models and sketches respetively are the mean squared errors (MSEs):

$$MSE_{\mathscr{P}}(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2) = \sum_i (\boldsymbol{\alpha}_{1i} - \boldsymbol{\alpha}_{2i})^2 \tag{5.4}$$

$$MSE_{\mathcal{M}}(\boldsymbol{m}_1, \boldsymbol{m}_2) = \sum_j \left(\boldsymbol{m}_{1j} - \boldsymbol{m}_{2j}\right)^2. \tag{5.5}$$

We can observe how a change from the original sketch is reflected in the models by plotting the MSE of the models (Equation 5.5) with respect to the MSE for the sketches (Equation 5.4). In this case, the MSE is calculated using the interpolated sketch (model) and the original sketch (model). The expectation is that as the interpolated sketch differs from the original sketch (in terms of MSE), the MSE between the models will also increase.

Some examples of this procedure are shown using projections of the higher dimensional models in Figure 5.6. Notice how every perturbation of the sketch results in a change in the model, which is exactly what we imply when we say that model is (locally) unique. Some example plots of model MSE vs. sketch MSE are shown in Figure 5.7.

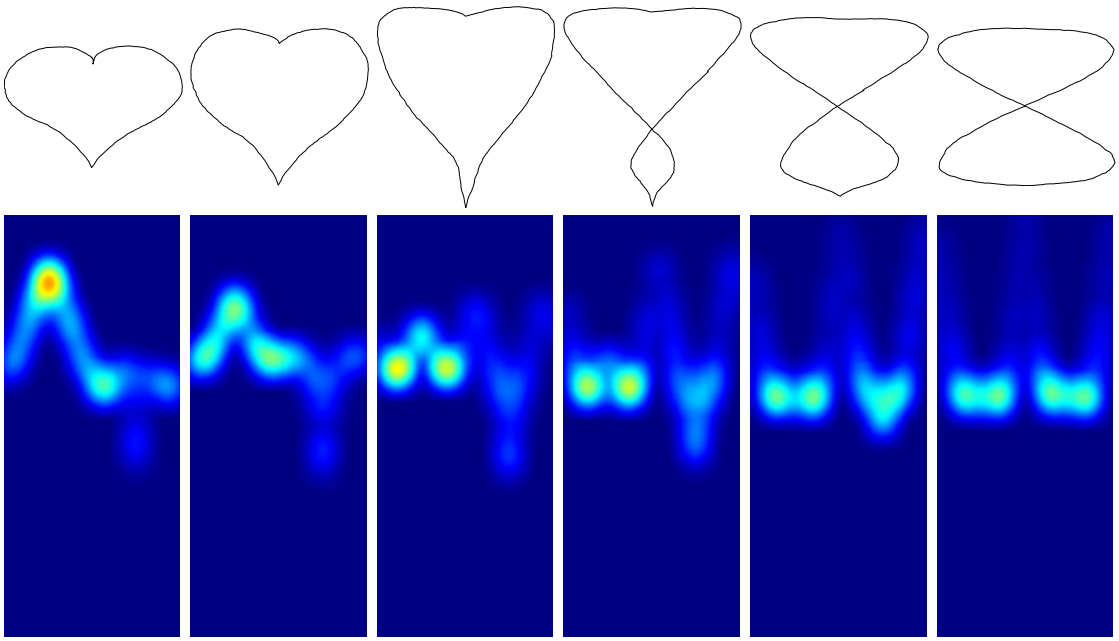


Figure 5.6: Shows the interpolation between a heart and a figure-8 (row 1). Then, the corresponding projections of the models onto the ρ-s plane are shown in row 2. Here, we can see that the first two shapes (and models) are more similar than the first and the last.

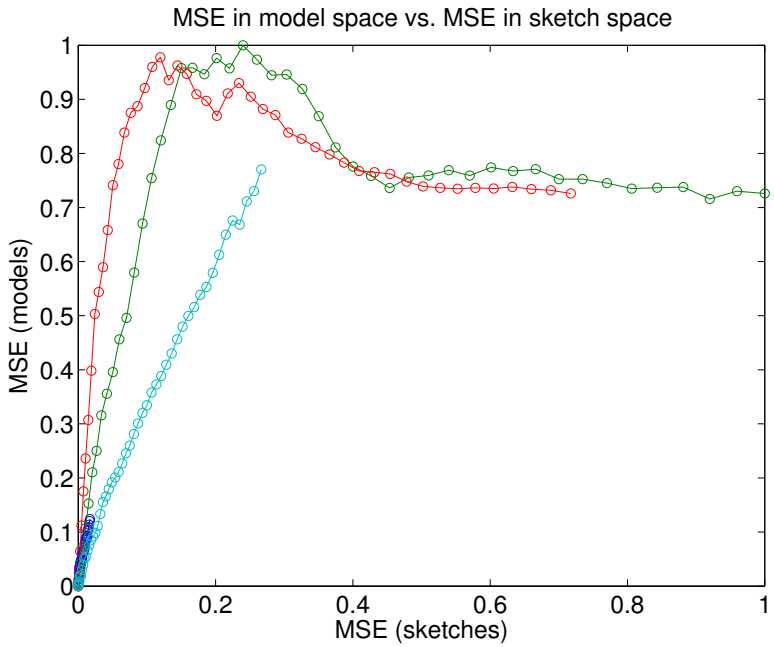


Figure 5.7: The plot shows how the MSE in the "space of sketches" is reflected in the "space of models" for the interpolation from one sketch to 4 different sketches. Initially, the MSE is null for both because the sketches and models are identical. However, as the sketches become different, the models also become increasingly different which is reflected in the examples shown here. The exact nature of the curve depends on both starting point and ending point in the "space of sketches."

5.5.2 Robustness

In the following experiment, we demonstrate the robustness of the accumulative framework. As in the previous section, we use the interpolation between two distinct sketches. While holding the model constant (i.e., let the model be constructed from the first in the interpolated sequence), let the login sketch vary. As the sketch differs, the accumulator peak changes too. Initially, the difference between the accumulator peaks is very little. As the sketch differs more and more, the accumulator peak becomes less distinctive, indicating less similarity

between sketches. Therefore, despite small perturbations (from the sketch producing the model) occurring, the accumulator will still exhibit a sufficiently larger peak, indicating a matching sketch. This implies that the matching procedure is intentionally fuzzy, which makes the system more robust.

A sequence of test shapes and the accumulators are shown in Figure 5.8.

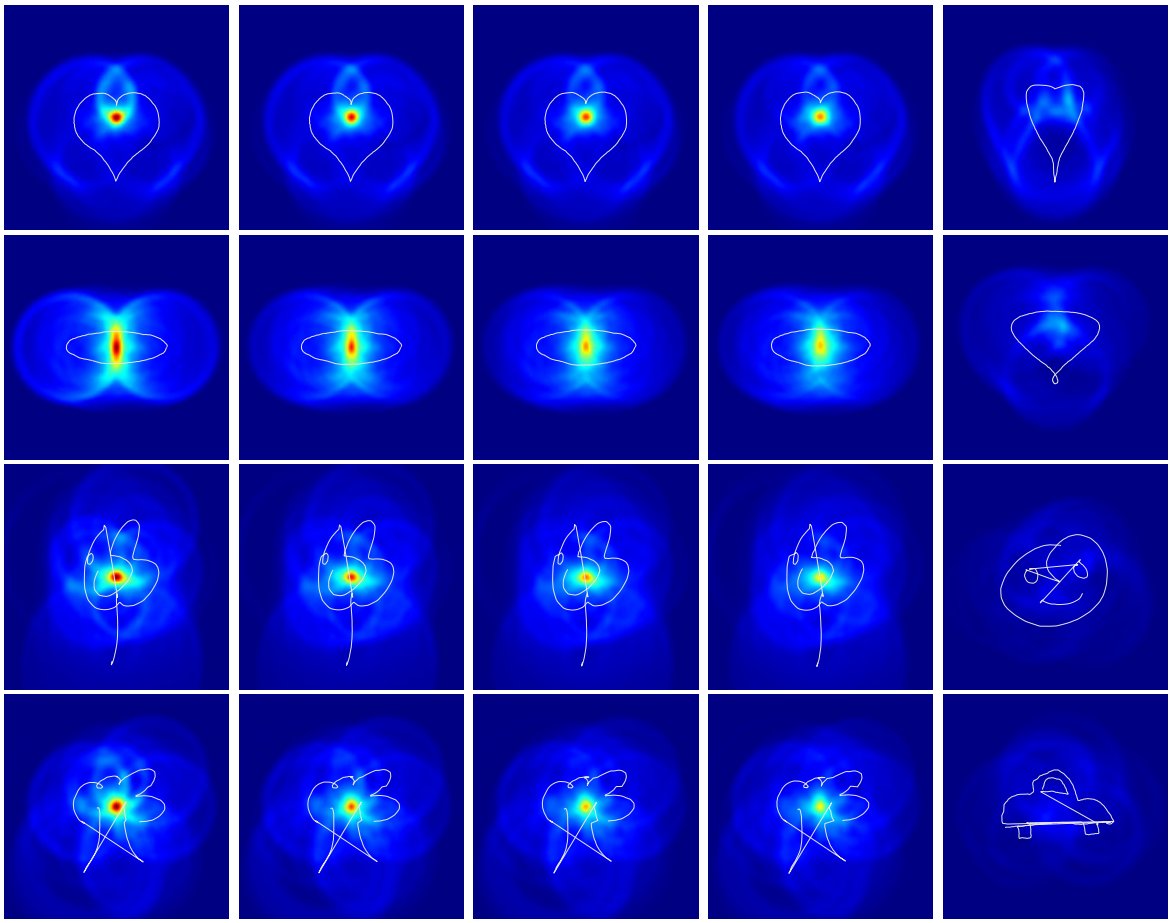


Figure 5.8: The test sketch is overlaid on the accumulator in order to show how the accumulator changes as the shape is deformed. Notice, how even deformed versions of the same sketch reveal a peak in the accumulator (columns 1–4). However, if the sketch is deformed too much the peak is diminished (column 5).

5.5.3 Synthetic Sketches

Using a set of parameterized shapes (with normalized scale), an ideal dataset is constructed (Figure 5.9). From these ideal shapes, random noise is added to the shape. The type of noise we are modeling is jitter, due to the shaking of the hand while drawing a sketch. Using this type of noise, the performance of the algorithm discussed in this paper is computed using four different feature combinations:

1. ρ and κ (Shape features)
2. ρ, κ, and θ (Shape and direction features)
3. ρ, κ, and b (Shape and pressure features)
4. ρ, κ, θ, and b (Shape, direction, and pressure features)

Jitter is modeled by low frequency, additive Gaussian noise. This type of noise is generated by lowpass filtering a sequence of independent Gaussian random variables with zero mean and some variance (σ^2). This low frequency noise is added to the spatial coordinates of the shape, and then the corresponding features are computed using the noisy signal.

A password model is constructed using three different noisy instances of a shape, and then tested on a total of 50 noisy shapes (10 of which are the same shape with simulated jitter). This procedure is performed for a total of five different shapes in order to compute FAR and FRR measures. By providing a different set of three sketches (of the same shape) to build each model, we can compute different FAR and FRR measures. Therefore, we repeatedly choose sets of three sketches to build each password model and compute an average FAR and average FRR. In this paper, this is how we will compare each method using different feature combinations.

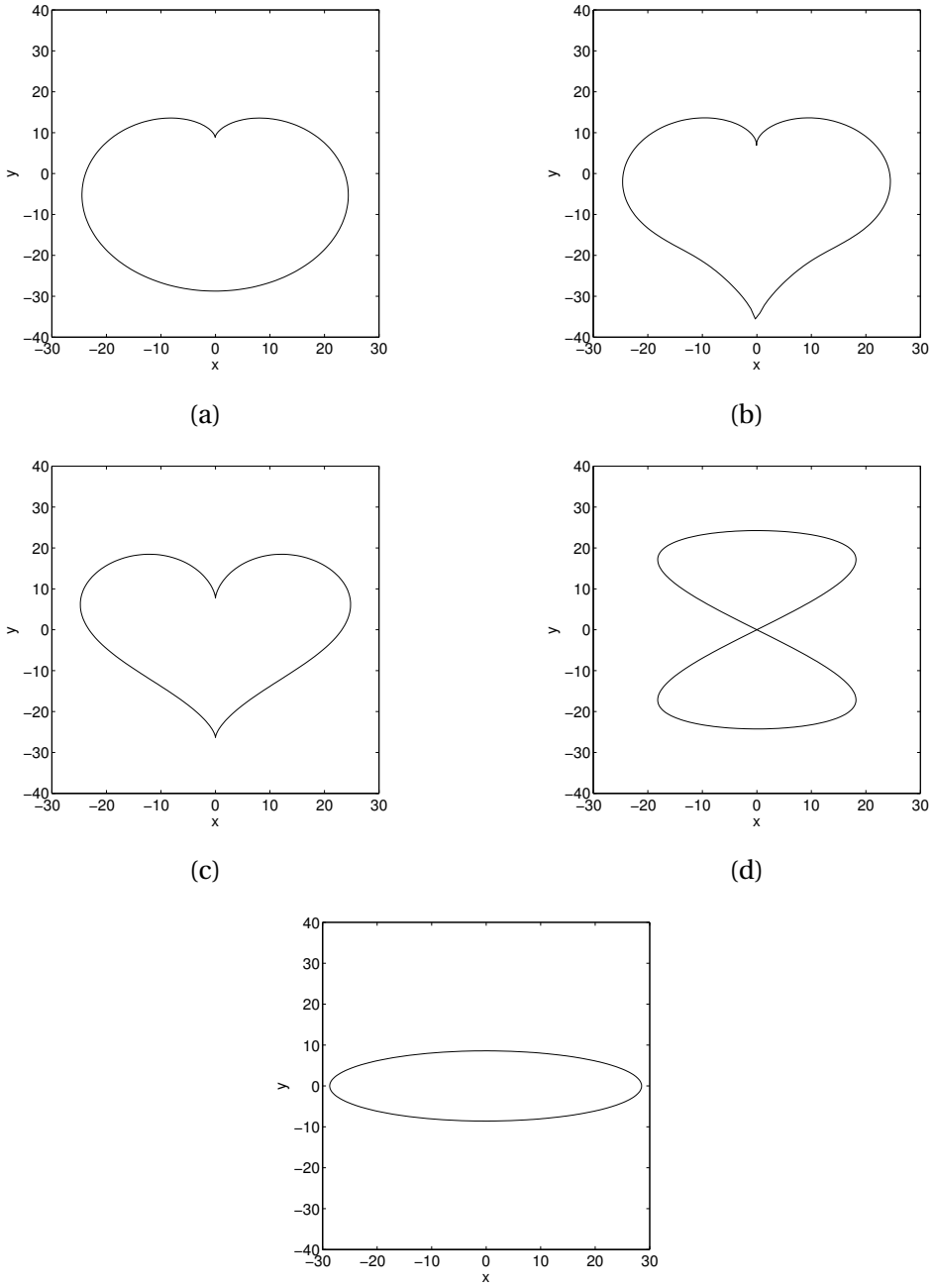


Figure 5.9: Set of synthetic sketches: (a) heart #1, (b) heart #2, (c) heart #3, (d) figure-8, and (e) ellipse

The performance curves in Figure 5.10 demonstrate that the generalized SKS method proposed for matching sketch-based passwords presented in the paper improves performance (at least for the synthetic data generated) from the original SKS approach. These results also show that it is much more robust under perturbations due to (low frequency) jitter. In principle, this also indicates the potential robustness to jitter from hand-drawn sketches.

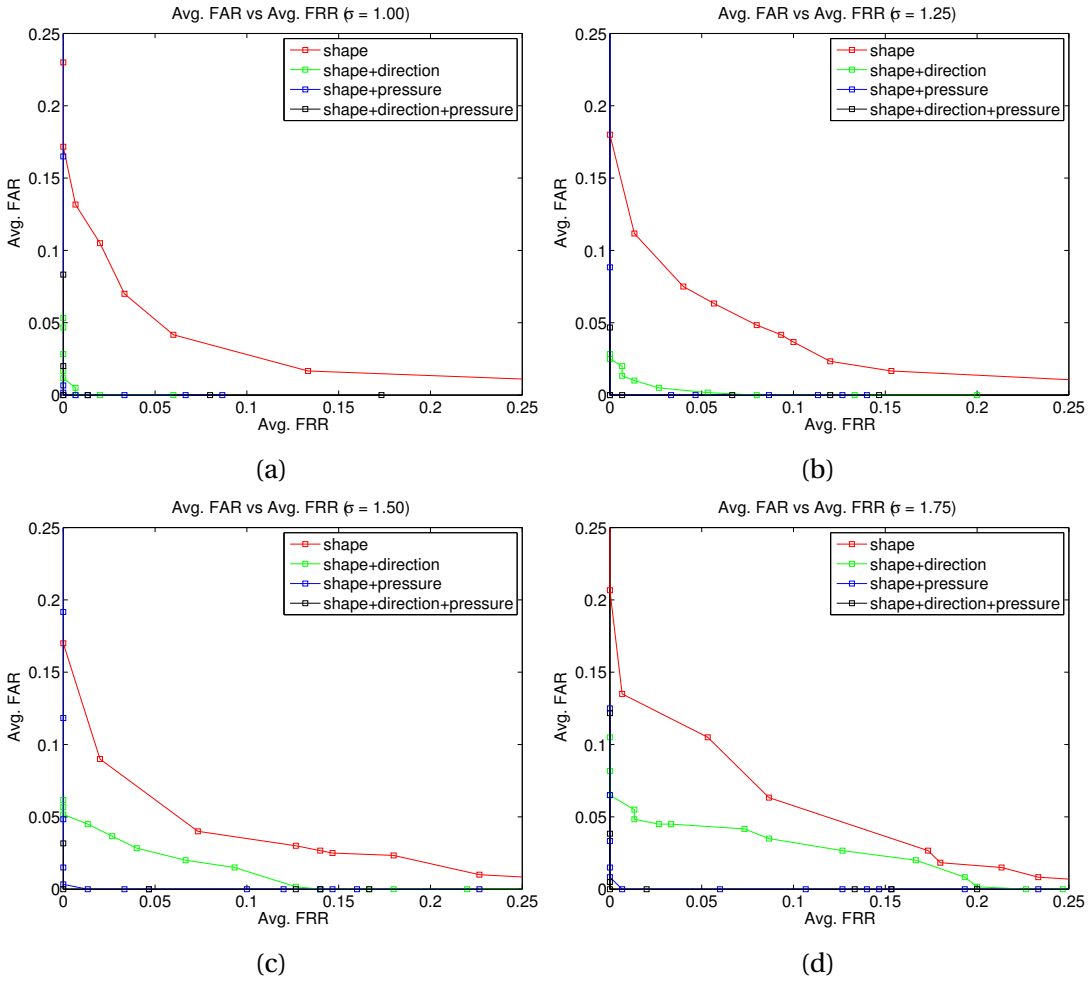


Figure 5.10: Comparison of average performance curves using different sets of features for different levels of jitter.

5.5.4 DooDB Database

In this set of experiments, we use the DooDB database [69, 68], which contains a set of finger-drawn doodles (or sketches) and pseudo-signatures. For the purposes of comparing our system with a state-of-the-art method, we outline our experiments in the same manner.

The DooDB database contains files which include both spatial and temporal information, which are used for positional, velocity, and acceleration type features. Since their data was collected using a device with a resistive touch screen and without any hardware for detecting pressure, they do not use pressure as a biometric feature. Therefore, to compare the systems on a level playing field, our system is tested on the DooDB database *without the biometric pressure*. We will demonstrate the advantage of using biometric pressure in Section 5.5.5.

The performance of our SKS-based method is compared with the method used in [68], which is a Dynamic Time Warping (DTW) based system. They test performance on both doodles (or sketches) and pseudo-signatures, and they report an EER for both random and skilled forgeries (with the expectation that skilled forgeries will have worse performance).

The performance measures used are the random forgery EER (denoted as EER_{rd}) and skilled forgery[9] EER (denoted as EER_{sk}) for both doodles and pseudo-signatures.

The results from [68] and our results are both reported in Table 5.1 for comparison. In [68], three different features sets are used with the DTW method. These features are ATVS-Pos, ATVS-Vel, and ATVS-Acc, which represent the spatial coordinates, velocity, and acceleration features respectively.

The results in Table 5.1 show that even *without biometric pressure* SKS is comparable to (better than in some cases) the DTW method. However, there are some important observations to note. First, SKS appears to perform slightly better than DTW on the doodles,

[9]A skilled forgery is defined as one in which the forger knows what the sketch looks like as well as its beginning and ending points

Table 5.1: Performance comparison between DTW and SKS on DooDB database, which included both doodles and pseudo-signatures. The best performance is achieved using the SKS-Bio, which is the approach presented in this paper.

		Doodles		**Pseudo-signatures**	
Method	**Session**	**EER**$_{rd}$	**EER**$_{sk}$	**EER**$_{rd}$	**EER**$_{sk}$
ATVS-Pos	1	2.7	28.0	3.5	28.6
ATVS-Vel	1	3.4	**26.7**	**1.6**	23.9
ATVS-Acc	1	4.5	28.1	2.2	**19.8**
SKS	1	**1.4**	28.0	**1.6**	23.3
ATVS-Pos	2	7.6	36.4	5.0	34.5
ATVS-Vel	2	6.3	**33.9**	**3.8**	29.7
ATVS-Acc	2	7.3	34.1	4.3	**25.0**
SKS	2	**3.8**	35.9	5.2	28.5

but not on the pseudo-signatures. Recognizing doodles (or sketches) was the objective of our approach, and we have achieved an improved results on the set of doodles. For the pseudo-signature, we still achieve comparable performance to the DTW approach, despite not designing the system with signatures in mind. Second, but probably most significant, is that our system is easily generalized to incorporate features such as biometric pressure as a property of a sketch. Since the DooDB database does not have pressure, we cannot accurately compare the two systems using pressure. However, we believe that biometric pressure significantly improves performance over those systems without it. Finally, the DTW approach tests both velocities and accelerations. In our experiments, we found that the improvement from velocity came more from the direction than the magnitude (or speed). Now, acceleration, which is a second derivative with respect to time, is a very noisy feature. In some cases, acceleration improves performance and in others acceleration worsens it.

5.5.5 Biometric Pressure

The following set of experiments aims to demonstrate the potential security benefit from adding biometric pressure to sketch-based password systems. A total of 100 sketches (with pressure information) from 5 different users were obtained using a Samsung Galaxy Note 10.1. Each of the 5 users provided 10 genuine sketches (i.e. 50 genuine sketches in total), and the remainder of the 100 sketches were skilled forgeries. Similar to the experiments performed on the DooDB database, we build a model using 3–5 example sketches (the first 3–5 sketches provided) and measure the verification performances (in terms of EER_{rd} and EER_{sk}) on the remaining sketches. The results are shown in Table 5.2.

Table 5.2: Performance comparison between the sketch-based password system with and without biometric pressure. Observe how the pressure makes an improvement in performance in almost all cases: all skilled forgery scenarios, and the 5 example case for random forgeries.

Method	**# Examples**	**EER**$_{rd}$	**EER**$_{sk}$
SKS	3	**2.5**	17.8
SKS-Press	3	3.5	**10.7**
SKS	4	**2.5**	16.6
SKS-Press	4	4.1	**3.7**
SKS	5	1.6	14.0
SKS-Press	5	**0.0**	**3.3**

The performance difference between SKS without pressure (SKS) and SKS with pressure (SKS-Press) is substantial; a more than 12% reduction in EER. The performance improvement for the random forgery scenario is less significant than that of the skilled forgery scenario. This is reasonable because in the random forgery case, the shape and directional components of the sketch are already mostly distinct. Thus, there is not much to improve upon. However, for skilled forgeries the shape and direction components are as similar as humanly possible.

Therefore, we can attribute the performances differences between SKS and SKS-Press to the biometric pressure. In every skilled forgery scenario tested: 3, 4, and 5 example sketches, the best performance is achieved by the addition of pressure.

CHAPTER 6

System Implementation

In this chapter, implementation using discrete sketches is discussed. Until now, the framework has been presented in a continuous sense, i.e. continuous sketches, continuous models, and continuous accumulators. In practice (e.g. examples from Chapter 5), discrete sketches must be used. Therefore, considerations which are specific to a discrete sampling are presented.

6.1 Sampling

Consider a discrete sampling (with k samples) of a continuous sketch, $\boldsymbol{\alpha}(s) = [x(s),\ y(s)]^T$, to be defined as:

$$\boldsymbol{\alpha}_d = \left\{\alpha_d^{(1)},\ \alpha_d^{(2)},\ \ldots,\ \alpha_d^{(k)}\right\}.$$

In general, the first discrete sample corresponds to continuous point with arc length $s = 0$.

Similarly, the k^{th} discrete sample corresponds to the continuous point with arc length $s = 1$ (without loss of generality).

Initially, the sampling is handled by the tablet or digitizer itself. In most real systems, the resolution of touch screens and other hardware is finite. Therefore, as a user physically draws a continuous sketch on the device, the sketch is sampled and thus discretized.

In practice, interrupt service routines are used to handle processing of sketch samples. There are two types of interrupts commonly used:

1. Timer-based (fixed frequency)

2. user interface-based (variable frequency)

Timer-based interrupts poll for the pen position, pen pressure, and etc. using a fixed frequency timer. Therefore, after a period of $T = \frac{1}{f}$ seconds the next sample is obtained. There are three cases that must be considered when using a timer-based interrupt: (1) a *new* sample point is obtained, (2) the same sample point as the previous is obtained, (3) no point is obtained.

Case (1) is where as the user is moving the pen, the interrupt service routine called and observes the change from the previous sample point. Next, assumes the pen is on the active surface of the device, but there is no movement by the pen (i.e. the current and previous samples are identical). In this case, one can either capture the redundant samples, or remove them. Lastly, case (3) is where the user has lifted the pen. Similar to case (2), the void samples may be retained or ignored.

UI-based interrupts only responds to the interactions from the user, e.g. pen ups, pen downs, and pen motions. The difference with UI-based sampling is that it is not handled at a fixed frequency, which means the variable sampling frequency is dependent on velocity of

the drawing process. This also means that it is possible to have an insufficient sampling[10] of the sketch, which means that some type of interpolation is required (see Section 6.5).

The primary differences between Timer-based and UI-based sampling are summarized in Table 6.1.

Table 6.1: Comparison between Timer and UI based interrupts.

description	**timer-based**	**UI-based**
frequency	fixed	variable
redundant samples	yes	no
absent samples	yes	no

In Equation 5.2, a continuous model is constructed from a continuous sketch. Even though a discrete sketch is used, it is possible to construct either a continuous or discrete model.

6.2 Continuous Model

The continuous model constructed from a discrete sketch is defined as:

$$m_{\boldsymbol{\alpha}}(\mathbf{v}) = \frac{1}{z}\sum_{i=1}^{k} \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}_d}^{(i)})^T \boldsymbol{\Sigma}^{-1}(\mathbf{v}-\mathbf{v}_{\boldsymbol{\alpha}_d}^{(i)})\right), \tag{6.1}$$

where $\mathbf{v}_{\boldsymbol{\alpha}_d}^{(i)}$ is the feature vector corresponding to the i^{th} discrete sample of the sketch $\boldsymbol{\alpha}_d$.

The primary difference between Equation 6.1 and Equation 5.2 is the summation over a finite number of samples, opposed to the integration over an infinite number of samples. Although a discrete sketch is used, the function is still continuous. The continuous model

[10] Insufficient sampling is also possible with timer-based interrupts if processing is too slow for the interrupt service routine.

resembles a kernel density estimator with a Gaussian kernel. The goal of a density estimator is to construct the best estimate of a continuous distribution using a finite number of samples from that distribution. Similarly, the goal here is to construct the continuous model from a finite set of sample points.

6.3 Discrete Model

Given a discrete sketch, a discrete model may also be constructed. The discrete model may be considered as a sampling of the continuous model. Consider a uniform sampling of $\mathscr{F}$ (the domain of $m_{\boldsymbol{a}}$), for every sample point, $\mathbf{v}_i$ there is a corresponding value $m_{\boldsymbol{a}}^{(i)} = m_{\boldsymbol{a}}(\mathbf{v}_i)$. A 2D example is depicted in Figure 6.1.

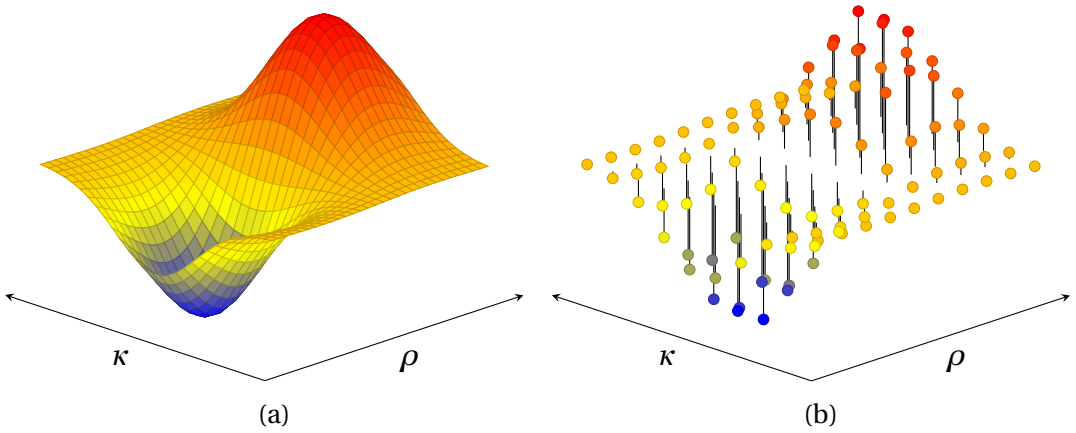


Figure 6.1: The illustration shows how the continuous model (a) is sampled in order to produce the discrete model (b).

In practice, a discrete model is actually used. Usually a sufficient number of samples are used to capture the overall structure of the model. In general, the normal sampling theorem

restrictions apply, i.e., the sampling frequency must be twice that of the largest frequency present in the model in order to reconstruct the signal. If the sampling rate is less the twice the large frequency and one attempts to recover a higher resolution model from subsampled version (maybe for reduced memory usage), alias will occur. Aliasing of this nature may cause problem in the accuracy of the recognition.

6.4 Feature Estimation

6.4.1 Curvature

Curvature in differential geometry is defined as the magnitude of the derivative with respect to s of the unit tangent vector [77]. That is, $\kappa(s) = |T'(s)| = |\boldsymbol{\alpha}''(s)|$, assuming the curve is unit speed. However, if the curve is not unit speed then the curvature of any plane curve is

$$\kappa(t) = \frac{x'(t)y''(t) - x''(t)y'(t)}{\left(x'^2(t) + y'^2(t)\right)^{\frac{3}{2}}}$$

where t denotes time. In either case, the curves are assumed to be continuous and twice differentiable. Both of which are only approximately true with discretely sampled sketches (i.e. discrete samples are assumed to be connected using by some continuous segment except for endpoints, and derivatives are approximate).

In practice, when working with discrete curves, curvature must be estimated. The general rule is that curvature at a point is the reciprocal of the radius of the osculating circle—the circle that best approximates the curve around that point (Figure 6.2). Determining the osculating circle at point q_i is performed (as in [45]) using the following algorithm (depicted in Figure 6.3):

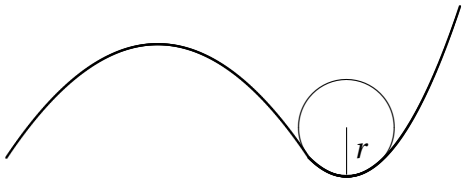


Figure 6.2: Osculating circle with radius r.

1. Determine k_b (k_f)—the length of the longest backward (forward) digital straight segment (DSS) from the point q_i.

2. Find the bisecting lines ℓ_b and ℓ_f of the line segments from q_{i-k_b} to q_i and from q_i to q_{i+k_f} respectively.

3. Compute the center of the osculating circle, c as the intersection of ℓ_b and ℓ_f.

4. Compute the radius of the osculating circle as the Euclidean distance between c and the point q_i.

5. Compute curvature as $\frac{1}{r}$.

The radius of the osculating circle is computed, like [41] (cited in [76]), as

$$r = \frac{s_1 s_2 s_3}{\sqrt{(s_1+s_2+s_3)(-s_1+s_2+s_3)(s_1-s_2+s_3)(s_1+s_2-s_3)}}$$

where $s_1 = \|q_i - q_{i-k_b}\|$, $s_2 = \|q_i - q_{i+k_f}\|$, and $s_3 = \|q_{i-k_b} - q_{i+k_f}\|$. This algorithm is illustrated in Figure 6.3.

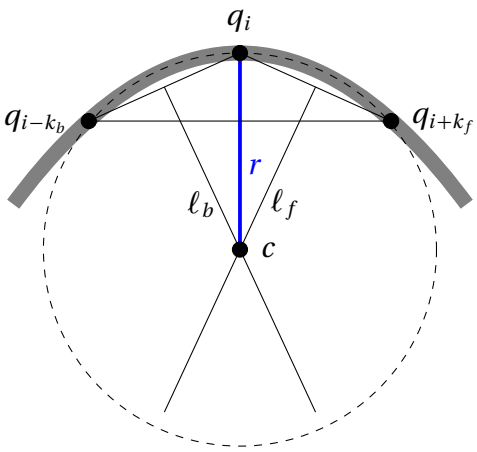


Figure 6.3: Estimation of the osculating circle at q_i. The radius r is shown as the line between the circumcenter c and the point q_i.

6.4.2 Tangent Direction

The tangent direction, θ, must also be estimated since discrete sketches are being used. One way to estimate the tangent direction, using a 3-point neighborhood about the point of interest i, is $\theta_i = \text{atan2}(y_{i+1} - y_{i-1}, x_{i-1} - x_{i+1})$. θ_i is a scalar value in the range $(-\pi, \pi]$, which represents the angle between the tangent vector and the x-axis at the i^{th} point on the sketch.

6.5 Interpolation

Thus far it is assumed that the sampling rate used to sample the continuous sketch is sufficient to accurately represent the original sketch. This might be true on an high performance desktop or laptop computers. However, implementing on a smart phone or tablet, whose processor and memory speeds are significantly slower compared to those in high performance computers, results in lower sampling rates. Therefore, the assumption of a sufficient sampling rate cannot be guaranteed, which makes interpolation necessary.

6.5.1 Linear

A trivial solution is to use a linear interpolant to determine the unknown points around the sketch. Linear interpolation uses a two point neighborhood since two points are necessary for determining the equation of a line. Given that $t \in [t_i,\ t_{i+1}]$, then the interpolation is computed using

$$f(t) = a_1 \left(\frac{t - t_i}{t_{i+1} - t_i} \right) + a_0$$

where

$$a_1 = -\frac{1}{2} f(t_i) + \frac{1}{2} f(t_{i+1})$$
$$a_0 = f(t_i)$$

and $\{(f(t_i), t_i),\ (f(t_{i+1}), t_{i+1})\}$ is the two point neighborhood of i^{th} discrete sample point on the sketch.

Linear interpolation is a quick and easy solution, but it is not very accurate when considering the nonlinearity of many sketches. More importantly, since curvature (i.e. a second derivative) is used in the recognition algorithm, this implies that linear interpolation only yields curvature information at points between two different interpolants. Therefore, a higher order interpolation is necessary. Table 6.2 provides information about various higher order interpolants for purposes of comparison.

Table 6.2: A comparison of different ordered interpolations.

interpolant	**first derivative**	**second derivative**	**curvature**
linear	constant	0	constant 0
quadratic	1^{st} order	constant	piecewise constant
cubic	2^{nd} order	1^{st}	piecewise linear
n^{th} order	$n-1$ order	$n-2$ order	piecewise $n-2$ order

6.5.2 Cubic

Cubic interpolation usually provides a better representation of the continuous sketch than linear interpolation, but one danger is possible overfitting. Also, curvature information exists at all interpolated points due to the use of a third order interpolant (twice differentiable). Cubic interpolation uses a 4 point local neighborhood (opposed to a 2 point neighborhood for linear interpolation) to determine the interpolant between the two center points in the neighborhood. The interpolant formula for $t \in [t_i,\ t_{i+1}]$ is

$$f(t) = a_3\left(\frac{t-t_i}{t_{i+1}-t_i}\right)^3 + a_2\left(\frac{t-t_i}{t_{i+1}-t_i}\right)^2 + a_1\left(\frac{t-t_i}{t_{i+1}-t_i}\right) + a_0$$

where

$$\begin{aligned}
a_3 &= -\frac{1}{2}f(t_{i-1}) + \frac{3}{2}f(t_i) - \frac{3}{2}f(t_{i+1}) + \frac{1}{2}f(t_{i+2}) \\
a_2 &= f(t_{i-1}) - \frac{5}{2}f(t_i) + 2f(t_{i+1}) - \frac{1}{2}f(t_{i+2}) \\
a_1 &= -\frac{1}{2}f(t_{i-1}) + \frac{1}{2}f(t_{i+2}) \\
a_0 &= f(t_i)
\end{aligned}$$

and $\{(f(t_{i-1}), t_{i-1}),\ (f(t_i), t_i),\ (f(t_{i+1}), t_{i+1}),\ (f(t_{i+2}), t_{i+2})\}$ represent the 4 point neighborhood of the i^{th} discrete point on the sketch.

6.6 Memory Speed Tradeoff

An interesting tradeoff that occurs in many software applications is the trade-off between the amount of memory used and speed. Typically, less memory usage required implies more computation, and thus a slower application. Similarly, the more memory that is used tends

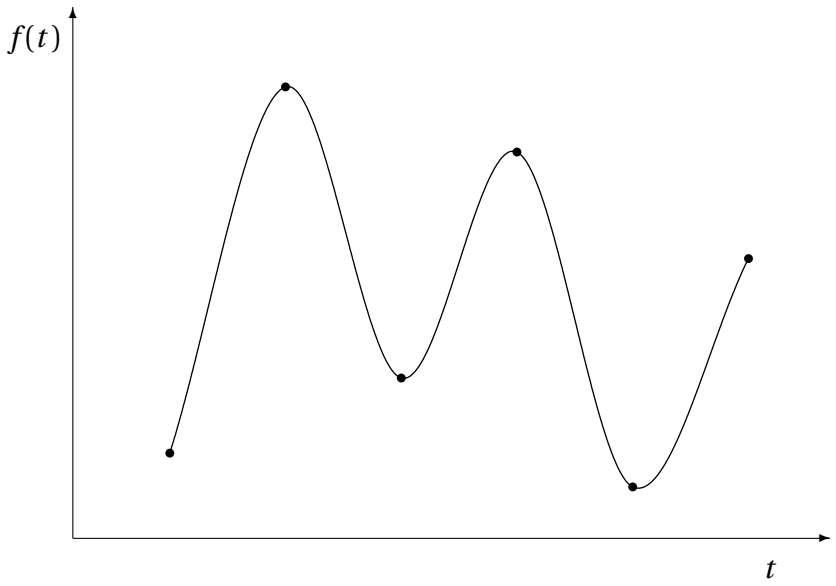


Figure 6.4: Cubic interpolation of a one parameter function.

to mean a faster application. However, the memory usage and speeds are not unbounded; physical limitations exists for both memory and computational speed. For memory the limitation is the amount of memory available for an application. For speed usually the limitation is algorithm implementation itself. This same tradeoff is also present in sketch-based password applications too.

6.6.1 More Memory and Less Time

The algorithm presented in Chapter 5 is the faster and more intuitive implementation. This approach requires the model to be precomputed and stored. Then, using the model and a login sketch, the accumulator is computed. How much storage does the model require? And how does this require relatively less time?

Since computing systems do not have infinite memory, the model being stored must be discrete. Therefore, we must sample the domain of the model and compute a value for each

point.

Observation 2 in Section 5.4 stated that the computational complexity scales linearly with the number discrete sample point along the sketch. This observation does not consider the memory requirements required to store the model. Assuming that the model is d-dimensional and N samples in each dimension, then the system must have N^d values available in order to store the model. Assuming that these values are double precision floating point values, then it will require $8N^d$ bytes. For example, a 5D model with 100 sample points in each dimension requires 80 gigabytes[11] (10^9 bytes) of storage for a model with double precision values (single precision requires 40 gigabytes). This is a lot of storage in Random Access Memory (RAM) even for modern computing systems.

After storing the model, the matching procedure treats the precomputed model as a lookup table (LUT). Computing the accumulator requires $p_\ell M$ memory accesses (or lookups), where p_ℓ is the number of sample points for the login sketch and M is the number of points in the accumulator.

6.6.2 Less Memory and More Time

There is a second implementation where the model is not precomputed and requires less memory, but more computation. The accumulator can be expressed using a single equation (Equation 6.2), which follows from Equation 5.2 and Equation 5.3.

$$\mathrm{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}}) = \frac{1}{z}\int_0^1\int_0^1 \exp\left(-\frac{1}{2}(\mathbf{v}_{\boldsymbol{\alpha}_\ell}(\hat{\tau},\hat{\mathbf{x}}) - \mathbf{v}_{\boldsymbol{\alpha}}(\tau))^T\boldsymbol{\Sigma}^{-1}(\mathbf{v}_{\boldsymbol{\alpha}_\ell}(\hat{\tau},\hat{\mathbf{x}}) - \mathbf{v}_{\boldsymbol{\alpha}}(\tau))\right)\,\mathrm{d}\tau\,\mathrm{d}\hat{\tau} \tag{6.2}$$

[11] Assuming model is uncompressed.

Similarly, there exists a discrete analog to Equation 6.2 which defined as:

$$\text{Acc}_{\boldsymbol{\alpha},\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}}) = \frac{1}{z}\sum_i\sum_j \exp\left(-\frac{1}{2}\left(\mathbf{v}^{(i)}_{\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}}) - \mathbf{v}^{(j)}_{\boldsymbol{\alpha}_d}\right)^T \boldsymbol{\Sigma}^{-1}\left(\mathbf{v}^{(i)}_{\boldsymbol{\alpha}_\ell}(\hat{\mathbf{x}}) - \mathbf{v}^{(j)}_{\boldsymbol{\alpha}_d}\right)\right) \tag{6.3}$$

This approach uses the login and reference sketch directly in order to computed the accumulator. Here, the model is computed but not stored. Therefore, the amount of memory required to stored the model is 0 bytes. However, the reduction in memory does not come without a price. When computing the accumulator (i.e. every time a person attempts to login), the model must be computed each and every time. Note that the complexity of building a model and logging in one time for either case (storing the model or not storing the model) is identical, but when multiple login attempts are necessary the former approach does not have recompute the model.

Table 6.3 summarizes the computational complexities for each approach.

Table 6.3: Summary of model, accumulator, and total computational complexities using each approach. "Approach 1" precomputes and stores the model, and "Approach 2" does not store the model.

Login Attempts	**Complexity**	**Approach 1**	**Approach 2**
	model	$p\mathcal{N}_p$	—
1	accumulator	$p_\ell M$	—
	model+acc.	$p\mathcal{N}_p + p_\ell M$	$p\mathcal{N}_p + p_\ell M$
	model	$p\mathcal{N}_p$	—
2	accumulator	$2p_\ell M$	—
	model+acc.	$p\mathcal{N}_p + 2(p_\ell M)$	$2(p\mathcal{N}_p + p_\ell M)$
	model	$p\mathcal{N}_p$	—
3	accumulator	$3p_\ell M$	—
	model+acc.	$p\mathcal{N}_p + 3(p_\ell M)$	$3(p\mathcal{N}_p + p_\ell M)$

CHAPTER 7

Human Factors Study

The work presented in this chapter primarily comes from [86].

In this chapter, the human-computer interaction study for sketch-based passwords is conducted using sufficiently large set of participants. This study was designed in collaboration with Dr. Jing Feng, who is a cognitive psychologist from the department of psychology at NC State University. IRB approval was obtained prior to conducting the study.

The primary goals of the study include: (1) constructing a sufficiently large and complete database of sketch-based passwords with biometric information, and (2) showing that sketch-based passwords are reproducible by humans, but secure from brute force, spoofing, shoulder surfing, and other attacks. To our knowledge, there is no existing database of sketches containing biometric information such as pressure. This general database will provide a framework for studying the variability of sketch-based passwords and adequately comparing the performance of methods using different features, algorithms, and hardware.

First, the BioSketch database is discussed, including: what and how data is collected. Then, a brief review of the metric (semi-metric) dissimilarity (similarity) measure used for the study is presented. Lastly, the results from the variability analysis and benchmark performance are provided.

7.1 Biometric Sketch Database

The BioSketch database is designed to capture the following local properties:

- Spatial Coordinates—(x, y)
- Pressure—b
- Time—t

Other local features may be derived directly from these properties, e.g. relative distances, curvature, velocity, acceleration, etc. Therefore, the BioSketch database is designed to be more general than the DooDB database [68].

7.1.1 System Design

The data is collected using a Samsung Galaxy Note 10.1 Android Tablet with a novel software application that we designed and implemented. The application layout includes five regions (labeled in Figure 7.1) for human-computer interaction:

1. Active Drawing Area
2. Instruction (or Dialog) Text
3. User Name Textbox

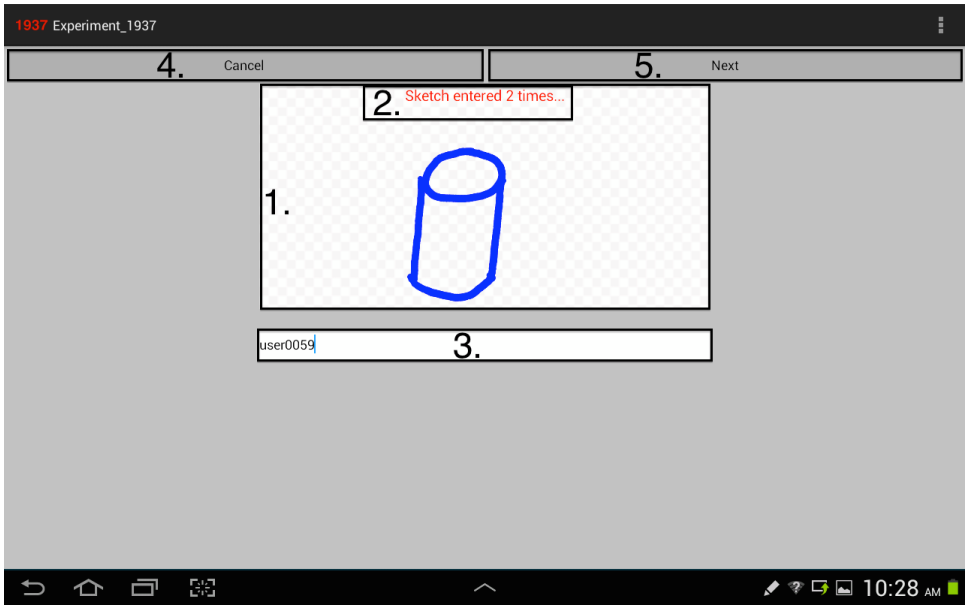


Figure 7.1: Screenshot of the application used to collect sketches from human participants for the BioSketch Database.

4. Cancel (or Clear) Button

5. Next (or Continue) Button

The active drawing area is where users provide the sketches, and the system records any pen-movements for the duration of the sketch. The instruction text is used only to provide users with directions to follow. The user name textbox is where users can input a username or identifier, which becomes part of the filename for the data stored by users. Lastly, the cancel and next buttons allow the user to start over or continue to the next sketch respectively.

7.1.2 Participant Study

The overall experimental design for collecting the data is shown in Figure 7.2. The experiment is designed to account for comfort, fatigue, stability, and short/long term recall. The

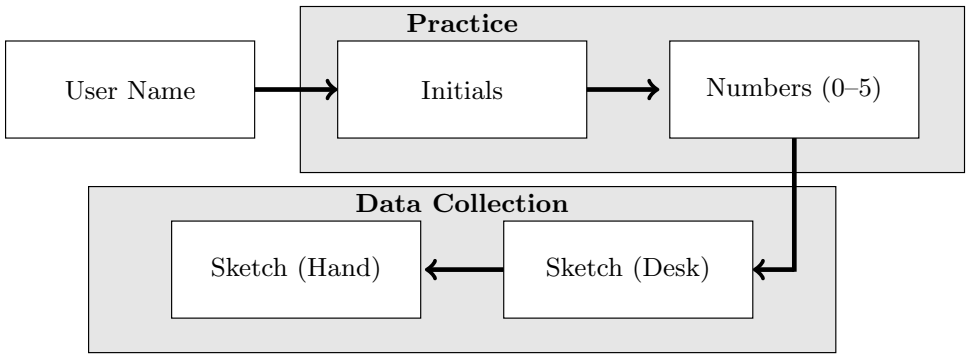


Figure 7.2: Block diagram of data collection experiment. The experiment consists of two phases: practice and data collection.

experiment consists of two nearly identical sessions, each composed of practice and data collection phases.

During session 1, the user is required to type his/her username for purposes of identification. The practice phase allows users to become comfortable drawing on the tablet. Every user is allowed the same amount of practice time; writing initials (5×) and numbers 0–5 (5× each). Sufficient practice also allows users that have not used a Samsung Galaxy Note 10.1 (or similar device) to become familiar with it for the purposes of this experiment. The data collection phase is where the sketches for the BioSketch database are obtained. Users are first asked to consider a sketch-based password.

For the purposes of this study, we asked that a sketch:

1. consists of a single uninterrupted stroke
2. consists of more than a single line segment or circle
3. not be a signature
4. be perceptively reproducible (to the user)

These conditions ensure that all sketches have similar complexity.

Although the matching algorithm we use is capable of extension to multiple strokes, reproducibility decreases with the number of strokes. Therefore, we allow the user to draw *almost* anything as long as they can draw it in a single stroke. Then, we asked that the sketch be more complex than a single line segment or a circle because these are similar to using "password" as a text-based password or "1234" as a PIN. We also asked users not to use their signature because signatures are known to have a higher intra-class variation than simple sketches. For example, consider a simple shape (e.g. a fish) and a signature. The complexity of each may be described by the number of self-intersections. The number of intersections for a fish is low (probably less than two intersections) and for a signature the number of intersections is much higher. Lastly, we simply ask that every user be sufficiently confident in their ability to reproduce the drawing. For example, we prefer that participants do not draw a Chinese character or symbol unless they already know Chinese. These restrictions only attempt to control and limit the scope of the experiment.

A total of 20 sketches are collected from every user; 10 drawn with the tablet lying flat on the desk (more stable), and 10 drawn with the tablet in hand (less stable). In order to account for the effects of fatigue, users are asked to draw at most 5 consecutive sketches.

Session 2 is practically identical to the first session. Users are required to refamiliarize themselves with the tablet using practice time, and then 20 more sketches are collected; 10 more stable and 10 less stable. However, this time they are asked to *recall* the sketch they produced during the first session. Users were asked to return for the second session between 2–10 days after session 1. The purpose for two sessions is to test both short term recall (session 1) and long term recall (session 2) of sketch-based passwords.

There are three testing scenarios: *genuine*, *random forgery*, and *skilled forgery*. The genuine case is where an instance of a user's sketch-based password is compared with another

instance of the same user's sketch-based password. The random forgery case is where a second users attempts to use his/her sketch to login as another user. For the skilled forgery case, a forger is assumed to have a priori knowledge of a genuine user's sketch-based password (except biometric information). Note that session 1 and session 2 described above are only for collecting sketches for genuine and random forgery cases (see examples in Figure 7.3); skilled forgeries are collected during a separate session from a different set of users.

In order to obtain skilled forgeries, selected forgers were presented with 10 examples of a particular user's sketch-based password as illustrated in Figure 7.4). Each example provides the forger with the sketch, start point, and end point. While forging a sketch, skilled forgers were allowed to re-draw any attempts that were poor replications, ensuring the sketches are actually "skilled."

Figure 7.3: Subset of sketches

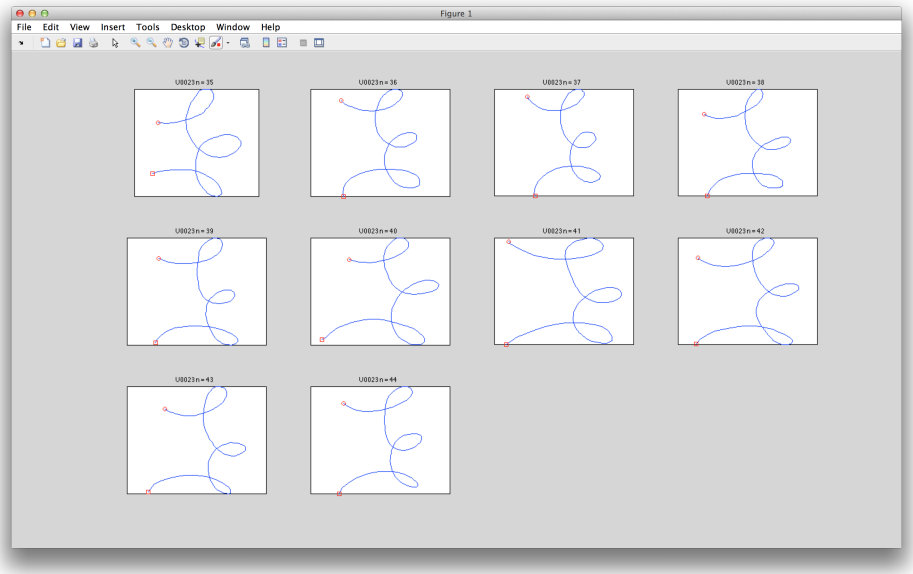


Figure 7.4: Example window shown to a forger for skilled forgeries.

7.1.3 Participants

The participants consisted of different genders, handedness, ages, and backgrounds. In total, there were 35 participants in session 1; 23 in session 2. Table 7.1 summarizes the number of male/female and left/right handed participants from each session. Additionally, another 15 participants were selected to provide skilled forgeries of sketches from session 1.

Table 7.1: Participant demographics.

Session	**Males**	**Females**	**Right**	**Left**
1	20	15	32	3
2	13	10	20	3

7.2 Similarity Measures

There are many measures, such as the Fréchet distance and SKS similarity measure, which have been used to quantify the differences between parameterized curves. The Fréchet and SKS measures (a.k.a. scores) are computed using a specific set of features: shape features (a distance ρ and curvature κ), trajectory (angle θ), biometrics (pressure b), and parameterization (arc length s), which may be computed from the measurements collected in Section 7.1. Here, we denote the features for a sketch $\boldsymbol{\alpha}$ by the vector $\mathbf{v}_{\boldsymbol{\alpha}}(s) = [\rho(s), \kappa(s), \theta(s), b(s), s]^T$.

The *Fréchet distance* [5, 6] is used for measuring the dissimilarity between parameterized curves, or in this case sketches. This distance is commonly described using the analogy of person walking a dog with a leash. The Fréchet distance represents the shortest leash required when the person walks forward along a designated path at some speed, and the dog walks forward along another path at another speed. The Fréchet distance is defined as

$$d_F(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \inf_{f,g} \max_{s} d(f \circ \mathbf{v}_{\boldsymbol{\alpha}}(s), g \circ \mathbf{v}_{\boldsymbol{\beta}}(s)) \tag{7.1}$$

where f and g represent any monotonic functions, which are used to reparameterize sketches $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively.

Simple K-space (SKS) [61], [96] was originally developed to match 2-dimensional shapes and images. It is well suited to that application because it is invariant to similarity transforms. The modification to use SKS for sketch-based passwords is straightforward (see [87] and Chapter 5 of this dissertation), and results in the following similarity measure (refer to Equation 6.2):

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \max_{\hat{\mathbf{x}}} \left\{ \int_0^1 \int_0^1 \delta(\mathbf{v}_{\boldsymbol{\alpha}}(s, \hat{\mathbf{x}}) - \mathbf{v}_{\boldsymbol{\beta}}(\hat{s}))\, \mathrm{d}s\, \mathrm{d}\hat{s} \right\} \tag{7.2}$$

where

$$\delta(\mathbf{z}) = \exp\left(-\frac{1}{2}\mathbf{z}^T\boldsymbol{\Sigma}^{-1}\mathbf{z}\right), \tag{7.3}$$

$\mathbf{v}_{\boldsymbol{\alpha}}(s,\hat{\mathbf{x}})$ and $\mathbf{v}_{\boldsymbol{\beta}}(s)$ represent the features of the sketches $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively; $\boldsymbol{\Sigma}$ is a matrix used to tradeoff security and robustness. The difference in notation is to distinguish the test sketch and the template sketch. The test sketch, in this case $\boldsymbol{\alpha}$, depends on a reference point $\hat{\mathbf{x}}$. The template $\boldsymbol{\beta}$ defines the reference as the center of gravity; although it can be arbitrary.

The likelihood L, in Equation 7.2, by definition is not symmetric. However, a new *symmetric* SKS measure is defined as follows:

$$L_{sym}(\boldsymbol{\alpha},\boldsymbol{\beta}) = L(\boldsymbol{\alpha},\boldsymbol{\beta}) + L(\boldsymbol{\beta},\boldsymbol{\alpha}), \tag{7.4}$$

which can be shown to be a semi-metric.

7.3 Benchmark and Analysis

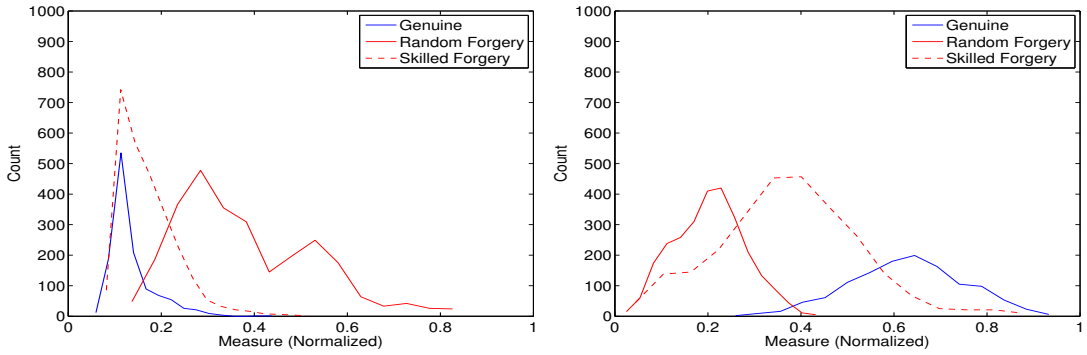
7.3.1 Variability Analysis

Next, we study inter-class and intra-class variations to better understand the human-computer interaction for sketch-based passwords. These variations are analyzed using both Fréchet and SKS measures. The variations are intended to show the differences between a “matching" and “non-matching” sketch. The differences between quantitative similarity and human perception of similarity motivate us to study theses variations.

The BioSketch database is composed of sketches from two sessions, each composed of two sets of sketches: session 1 with tablet flat on the desk (D1), session 1 with tablet in hand (H1), session 2 with tablet flat on the desk (D2), and session 2 with tablet in hand (H2). For

each of these sets, we histogram the scores for genuine sketches, random forgeries, and skilled forgeries.

The histograms (Figure 7.5) are used to illustrate the differences between genuine sketches, random forgeries, and skilled forgeries for D1; histograms for H1, D2, H2 are not shown, but are very similar to those shown above. Genuine scores are computed using $d_F(\boldsymbol{\alpha}, \boldsymbol{\beta})$ and $L_{sym}(\boldsymbol{\alpha}, \boldsymbol{\beta})$ respectively, where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are two distinct sketches drawn by the same person. Random forgery scores are also computed using Equations 7.1 and 7.4, but $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are sketches drawn by different users (from session 1 and session 2). $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are also drawn by different individuals for skilled forgeries, however, skilled forgers attempt to accurately replicate a genuine user's sketch using a priori knowledge of the sketch-based password.



(a) Fréchet distance (b) SKS

Figure 7.5: The plots show the genuine, random forgery, and skilled forgery distributions for D1. Recall that the Fréchet distance (SKS) is a dissimilarity (similarity) measure, which means that genuine sketches have lower (higher) scores.

In Figure 7.5, we are interested in the random forgery overlap, skilled forgery overlap, and histogram features (e.g. tails, peaks, and variance). The amount of overlap between the genuine scores and random forgery scores is very small for both Fréchet and SKS, and the overlap between the genuine and skilled forgeries is greater. When comparing the Fréchet

distance and SKS for D1 (and other sets) it is evident that the amount of overlap is generally less for SKS. Now, consider the features of the histograms. The histogram of the genuine scores for d_F exhibits a tail, which indicates the presence of outliers in the data. The peaks in the histograms indicate the most probable scores for genuine sketches and forgeries. Notice that peaks for the genuine sketches and skilled forgeries occur nearly at the same location using d_F, and they are further apart using L_{sym}. Finally, compare the variances of the genuine and forgery histograms: the genuine variance for the Fréchet distance is smaller than that for SKS, which indicates that Fréchet has a more restricted[12] definition of similar than SKS.

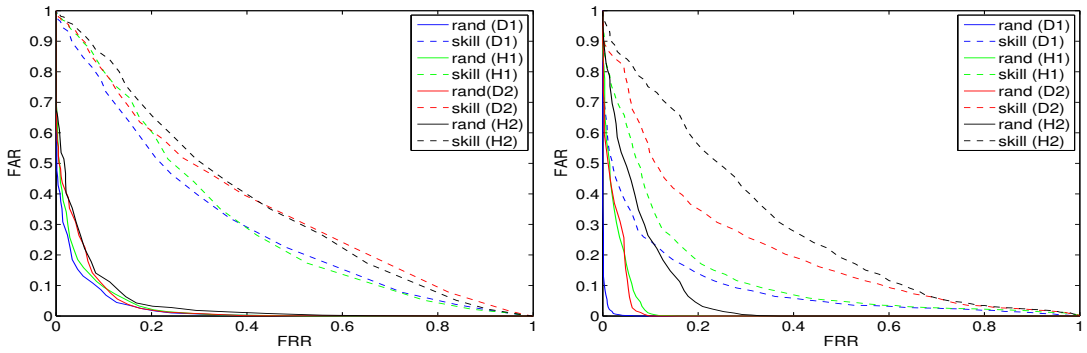
A fundamental problem with comparing and analyzing sketch-based passwords is knowing the ground truth. In this paper, we assume that if a person actually drew the sketch, then it is similar to all other sketches he/she provided. However, people are not perfect when it comes to drawing. In fact, from the histograms we observed the presence of outliers in the set of presumably genuine sketches. Therefore, the assumption that human behavior can be used as a gold standard is not ideal, but it is the method most commonly used in various pattern recognition problems.

7.3.2 Benchmark Performance

In most systems, performance is measured by the false acceptance rate (FAR)—percentage of forgeries wrongly considered to be genuine—versus the false rejection rate (FRR)—percentage of genuine sketches considered to be a forgery. Therefore, the performance curves are shown for both Fréchet and SKS methods (Figure 7.6).

Notice the differences between the performance curves using the Fréchet distance and SKS. A visual comparison of the curves reveals that in almost all cases SKS outperforms the

[12] SKS may be more robust than the Fréchet distance because it has broad definition of *similar* and a more narrow definition of *different*.



(a) Fréchet distance (b) SKS

Figure 7.6: FAR vs. FRR curves using the Fréchet distance and SKS.

Fréchet distance.

In order to better compare performance, Table 7.2 shows the random and skilled equal error rates (EERs)—the rate at which FAR is equal to FRR—for each method using the four sets of sketches (D1, H1, D2, H2). The results show as much as a 7% reduction in EER between the Fréchet distance and SKS using random forgeries and a 17% reduction for skilled.

Table 7.2: Summary of EER (%) performances using the Fréchet distance and SKS. Bold face indicates best performances.

Session	Method	EER (Random)	EER (Skilled)
D1	Fréchet	9.05	34.36
	SKS	**1.74**	**16.75**
H1	Fréchet	9.76	34.73
	SKS	**6.85**	**19.10**
D2	Fréchet	9.97	39.55
	SKS	**5.72**	**27.86**
H2	Fréchet	**11.22**	39.96
	SKS	14.57	**34.52**

7.3.3 Survey

After collecting the sketches from each user, participants filled out a survey regarding the proposed graphical password system and human perceptions. The survey included questions regarding user comfort (when drawing on a tablet), consistency (as perceived by the user), and usability. The user indicated a number between one and ten, where one represents very little (if any at all) comfort, consistency, or usability and ten represents the most comfort, consistency, or usability. Table 7.3 provides the minimum, maximum, mean, and median of all user responses.

Table 7.3: Survey Statistics.

Question	**Min**	**Max**	**Mean**	**Median**
Comfort	5.0	10.0	9.2	10.0
Consistency	3.0	10.0	7.6	8.0
Usability	1.0	10.0	8.2	9.0

CHAPTER 8

Conclusions and Future Work

8.1 Conclusions

The system that was presented in this dissertation built a foundation upon earlier applications in shape and signature recognition, however, the major differences (besides the recognition algorithm) are: (1) minimal training (3–5 examples), (2) semi-supervision, and (3) one representative model. The novelty in this system is primarily in the recognition framework. This framework—generalized SKS—is critical for recognizing sketch-based passwords that incorporate biometric information, such as pressure. A unique analytical approach using fundamentals from manifold theory and differential geometry provided a way to demonstrate how SKS is capable of fine tuned control of the tradeoff between security and usability in this application.

SKS was compared with two state-of-the-art dissimilarity measures: DTW and the Fréchet

distance. In general, SKS achieves similar performance to DTW, *without using any biometrics*. However, SKS is more general than DTW. The additional restrictions, such as boundary conditions and monotonic constraints, that DTW impose on the optimization problem makes DTW less robust. SKS, which does not impose such constraints, is free to accumulate evidence of similarity between a sketch and a model. The addition of the biometric pressure was shown to increase the level of performance by more than 12%.

When comparing SKS with the Fréchet distance, it was shown that overlap in the genuine and forgery (random and skilled) score distributions are reduced when using SKS. These score distributions demonstrate that SKS is more robust than the Frèchet distance when it comes to natural variations due to the drawing process.

The bottom line result from the work presented is that 1) people are capable of reproducing both spatial and biometric components with sufficient accuracy and precision and 2) SKS is robust recognition framework well suited to the application of sketch-based passwords with biometric information.

8.2 Future Work

The are many extensions to the work presented in this dissertation, including smoothing parameter estimation, dimensionality reduction methods, and applications in higher spatial dimensions (opposed to higher dimensions of the model). Since SKS is similar to a kernel density estimator, a natural extension to the proposed method is to estimate the covariance matrix used for smoothing. There are some advantages to estimating this parameter including reducing the number of manually chosen algorithm parameters and estimating may better account for feature dependencies.

One possible drawback to the SKS approach discussed is the number of features used to

construct the model (i.e. model construct requires the sampling of a 5 dimensional feature space), which impact the computational complexity. This may be alleviated by using a subspace approximations, such as PCA or KPCA. Reducing the dimensionality may improve computational complexity and speed of the matching procedure, and due to the nature of principal components performance may even improve. However, this must be experimentally shown.

Lastly, we hope to extend this work to contours in 3D (e.g. gesture recognition). Using Microsoft's Kinect (or any RGB-D camera), a gesture may be provided by tracking the hand(s), joints, or limbs of an individual in 3D space. Therefore, an extension of SKS may be applied in a similar way to recognizing the contours made by the tracking path(s).

REFERENCES

[1] Intuos pro medium. `http://www.wacom.com/en/us/creative/intuos-pro-m`. Accessed: 06-15-2014.

[2] Samsung galaxy note 10.1. `http://www.samsung.com/us/mobile/galaxy-note/GT-N8013EAVXAR`. Accessed: 06-15-2014.

[3] Handwriting recognition with hidden markov models and grammatical constraints. *Proc. of the 4th Int'l Conf. on Frontiers in Handwriting Recognition*, pages 195–205, 1994.

[4] S. Abbasi, F. Mokhtarian, and J. Kittler. Curvature scale space image in shape similarity retrieval. *Multimedia Systems*, 7:467–476, 1999.

[5] P. K. Agarwal, H. K. Avraham, H. Kaplan, and M. Sharir. Computing the discrete freéchet distance in subquadratic time. *Proc. of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 156–167, 2013.

[6] H. Alt and H. Godau. Computing the fréchet distance between two polygonal curves. *Int'l Journal of Computational Geometry and Applications*, 5(1–2):75–91, 1995.

[7] K. Arbter. Affine-invariant fourier descriptors. In J. C. Simon, editor, *From Pixels to Features*, pages 153–164. Elsevier Science, 1989.

[8] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger. Applications of affine-invariant fourier descriptors to recognition of 3-d objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):640–647, 1990.

[9] D. Ballard. Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[10] S. O. Belkasim, M. Shridhar, and M. Ahmadi. Pattern recognition with moment invariants: a comparative study and new results. *Pattern Recognition*, 24(12):1117–1138, 1991.

[11] R. Bellman. *Dynamic Programming*. Princton University Press, 1957.

[12] E. Belogay, C. Cabrelli, U. Molter, and R. Shonkwiler. Calculating the hausedorff distance between curves. *Information Processing Letters*, 64, 1997.

[13] S. Belongie, J. Malki, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(24):509–522, 2002.

[14] N. Bennett, R. Burridge, and N. Saito. A method to detect and characterize ellipses using the hough transform. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(7):652–657, 1999.

[15] M. I. Bernstein. Computer recognition of on-line, hand-written characters. Memorandum RM-3753-ARPA, The RAND Corporation, Santa Monica, California, October 1964.

[16] R. Biddle, S. Chiasson, and P. C. Van Oorschot. Graphical passwords: learning from the first twelve years. *ACM Computing Surveys*, 44(4):1–25, 2012.

[17] R. L. Bishop and S. I. Goldberg. *Tensor analysis on manifolds*. Dover Publications, 1980.

[18] F. L. Bookstein. Linear machinery for morphological distortion. *Computers and Biomedical Research*, 11(5):435–458, 1978.

[19] F. L. Bookstein. The measurement of biological shape and shape change. In *Notes in Biomathematics*, volume 24. Springer-Verlag, 1978.

[20] F. Casacubera, E. Vidal, and H. Rulot. On the Metric Properties of Dynamic Time Warping. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 35:1631–1633, 1987.

[21] S. Chiasson, A. Forget, R. Biddle, and P. C. Van Oorschot. Influencing users towards better passwords: Persuasive cued click-points. *British Computer Society Conference on Human-Computer Interaction*, 1:121–130, 2008.

[22] S. Chiasson, P. C. Van Oorschot, and R. Biddle. Graphical password authentication using cued click points. *12th European Symposium on Research in Computer Society*, pages 359–374, 2007.

[23] S-B. Cho. Neural-network classifiers for recognizing totally unconstrained handwritten numerals. *IEEE Trans. on Neural Networks*, 8:43–53, 1997.

[24] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2nd edition, 1991.

[25] C. Dalitz, C. Brandt, S. Goebbels, and D. Kolanus. Fourier descriptors for broken shapes. *EURASIP Journal on Advances in Signal Processing*, 2013(1), 2013.

[26] R. Davies, C. Twining, and C. Taylor. *Statistical models of shape: optimization evaluation*. Springer, 2008.

[27] D. Davis, F. Monrose, and M. K. Reiter. On user choice in graphical password schemes. *Proc. of the 13th USENIX Security Symposium*, pages 151–164, 2004.

[28] A. De Angeli, L. Coventry, G. Johnson, and K. Renaud. Is a picture really worth a thousand words? exploring the feasibility of graphical authentication systems. *Int'l Journal of Human Computer Studies*, 63(1–2):128–152, 2005.

[29] R. Dhamija and A. Perrig. Déjà vu: A user study using images for authentication. *Proc. of the 9th USENIX Security Symposium*, pages 45–58, 2000.

[30] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley & Sons, 2nd edition, 2001.

[31] G. Dudek and J. K. Tsotsos. Shape representation and recognition from multiscale curvature. *Computer Vision and Image Understanding*, 68(2):170–189, 1997.

[32] P. Dunphy and J. Yan. Do background images improve "draw a secret" graphical passwords? *Proc. of the 14th ACM Conf. on Computer and Communications Security*, pages 36–47, 2007.

[33] M. Faundez-Zanuy. Signature Recognition State-of-the-art. *IEEE Aerospace and Electronic Systems Magazine*, pages 28–32, July 2005.

[34] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[35] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 2nd edition, 1987.

[36] M. Fuandez-Zanuy. Biometric Security Technology. *IEEE Aerospace and Electronic Systems Magazine*, 21(6):15–26, June 2006.

[37] H. Gao, X. Guo, X. Chen, L. Wang, and X. Liu. Yet another graphical password strategy. *Proc. of the Annual Computer Security Applications Conference*, pages 121–129, 2008.

[38] J. Goldberg and J. Hagman. Doodling our way to better authentication. *Proc. of the ACM Conference on Human Factors in Computing Systems*, pages 868–869, 2002.

[39] D. S. Guru and H. N. Prakash. Online Signature Verification and Recognition: An Approach Based on Symbolic Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):1059–1073, June 2009.

[40] M. Hamdani, A.E.-D. Mousa, and H. Ney. Open vocabulary arabic handwriting recognition using morphologycal decomposition. *Int'l Conf. on Document Analysis and Recogntion*, pages 280–284, 2013.

[41] J. Harris and H. Stocker. *Handbook of mathematics and computational science*. Springer-Verlag, 1998.

[42] S. Haykin. *Neural networks and learning machines*, chapter 4. Prentice Hall, 3rd edition, 2008.

[43] N. M. Herbst and C. N. Liu. Signature verification method and apparatus utilizing both acceleration and pressure characteristics. *U.S. Patent 4 128 829*, 1978.

[44] N. M. Herbst and H. Morrisey. Signature verification method and apparatus. *U.S. Patent 3 983 535*, 1976.

[45] S. Hermann and R. Klette. A comparative study on 2D curvature estimators. *Proc. of the Int'l. Conf. on Computing: Theory and Applications*, pages 584–589, 2007.

[46] G. F. Houle, D. B. Aragon, R. W. Smith, M. Shridhar, and D. Kimura. A multi-layered corroboration based check reader. In J. J. Hull and S. L. Taylor, editors, *Document Analysis System II*, pages 137–174. World Scientific, 1998.

[47] J. Hu, M. K. Brown, and W. Turin. Hmm based on-line handwriting recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(10), 1996.

[48] J. Hu, Lim S. G., and M. K. Brown. Writer independent on-line handwriting recognition using an hmm approach. *Pattern Recognition*, 33(1):133–147, 2000.

[49] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. on Information Theory*, 8(2):179–187, 1962.

[50] A. K. Jain, F. D. Griess, and S. D. Connell. On-line signature verification. *Pattern Recognition*, 35:2963–2972, 2002.

[51] A. K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *IEEE Trans. on Circuits and Systems for Video Technology*, 14(1):4–20, Jan. 2004.

[52] I. Jermyn, A. Mayer, F. Monrose, M. Reiter, and A. Rubin. The design and analysis of graphical passwords. *Proc. of the 8th USENIX Security Symposium*, pages 1–14, 1999.

[53] R. S. Kashi, Jianying Hu, and W. L. Nelson. On-line handwritten signature verification using hidden markov model features. *Proc. of the 4th Int'l. Conf. on Document Analysis and Recognition*, 1:253–257, 1997.

[54] D. G. Kendall. The diffusion of shape. *Advances in Applied Probability*, 9(3):428–430, 1977.

[55] D. G. Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin London Math. Soc.*, 16(2):81–121, 1984.

[56] D. G. Kendall, D. Barden, T. K. Carne, and H. Le. *Shape and shape theory*. John Wiley & Sons, 1999.

[57] M. Khalid, H. Mokayed, R. Yusof, and O. Ono. Online signature verification with neural networks classifier and fuzzy inference. *3rd Asia Int'l Conf. on Modeling & Simulation*, pages 236–241, 2009.

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

[59] A. Kosmala, J. Rottland, and G. Rigoll. An investigation of the use of trigraphs for large vocabulary cursive handwriting recognition. *IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing*, pages 3373–3376, 1997.

[60] K. Krish. *An accumulative framework for object recognition.* PhD thesis, North Carolina State University, 2009.

[61] K. Krish, S. Heinrich, W. E. Snyder, H. Cakir, and S. Khorram. Global registration of overlapping images using accumulative image features. *Pattern Recognition Letters*, 31:112–118, 2010.

[62] A. Levinshtein, C. Sminchisescu, and Sven Dickinson. Optimal contour closure by superpixel grouping. *Proc. of the 11th European conf. on Computer Vision*, pages 480–493, 2010.

[63] J. S. Lew. An improved regional correlation algorithm for signature verification which permits small speed changes between handwriting segments. *IBM Journal of Research and Development*, 27(2):181–185, March 1983.

[64] C. Liu, F. Yin, Q. Wang, and D. Wang. Icdar 2011 chinese handwriting recognition competition. *Int'l Conf. on Document Analysis and Recognition*, pages 1464–1469, 2011.

[65] Z.-Q. Liu, J. Cai, and R. Buse. Handwriting Recognition. Soft Computing and Probabilistic Approaches. In *Studies in Fuzziness and Soft Computing*, volume 133. Springer-Verlag, 2003.

[66] J. Makhoul, T. Starner, R. Schartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. *Proc. of the Int'l Conf. on Acoustics, Speech, and Signal Processing*, pages v125–v128, 1994.

[67] J. Martin and J.-B. Durand. Automatic handwriting gestures recognition using hidden markov models. *Proc. of the 4th IEEE Int'l Conf. on Automatic Face and Gesture Recognition*, pages 403–409, 2000.

[68] M. Martinez-Diaz, J. Fierrez, and J. Galbally. The DooDB Graphical Password Database: Data Analysis and Benchmark Results. *IEEE Access*, 1:596–605, 2013.

[69] M. Martinez-Diaz, J. Fierrez, C. Martin-Diaz, and J. Ortega-Garcia. DooDB: A Graphical Password Database Containing Doodles and Pseudo-Signatures. *12th Int'l. Conf. on Frontiers in Handwriting Recognition*, pages 339–344, 2010.

[70] M. M. Mehdi. Optimized word segmentation for the word based cursive handwriting recognition. *2013 European Modelling Symposium*, pages 299–304, 2013.

[71] O. R. Mitchell and T. A. Grogan. Global and partial shape discrimination for computer vision. *Optical Engineering*, 23:484–491, 1984.

[72] D. S. Moore and G. P. McCabe. *Introduction to the Practice of Statistics*. W. H. Freeman, 3rd edition, 1999.

[73] M. Munich and P. Perona. Continuous Dynamic Time Warping for Translation Invariant Curve Alignment with Applications to Signature Verification. *Proc. of the 7th International Conference on Computer Vision*, September 1999.

[74] E. Nakashima. U.s. notified 3,000 companies in 2013 about cyberattacks, March 2014. Accessed: 06-18-2014.

[75] V. S. Nalwa. Automatic on-line signature verification. *Proc. 3rd Asian Conf. Computer Vision*, 1:10–15, 1997.

[76] T. P. Nguyen and I. Debled-Rennesson. Curvature estimation in noisy curves. *Proc. of the 12th Int'l Conf. on Computer Analysis of Images and Patterns*, pages 474–481, 2007.

[77] J. Opera. *Differential geometry and its applications*, chapter 1, page 17. Pearson Education, 2007.

[78] PASSFACES CORPORATION. The science behind passfaces. `http://www.passfaces.com/published/The%20Science%20Behind%20Passfaces.pdf`, 2009.

[79] E. Persoon and K. Fu. Shape discrimination using fourier descriptors. *IEEE Trans. Systems, Man and Cybernetics*, 7(3):170–179, 1977.

[80] H. Pesch, J. Hamdani, M. andForster, and H. Ney. Analysis of preprocessing techniques for latin handwriting recognition. *Int'l Conf. on Frontiers in Handwriting Recognition*, pages 280–284, 2012.

[81] J. Picone. Continuous speech recognition using hidden markov models. *IEEE Acoustics, Speech, and Signal Processing Magazine*, pages 26–41, July 1990.

[82] R. Plamondon and G. Lorette. Automatic Signature Verification and Write Identification – The State of the Art. *Pattern Recognition*, 22(2):107–131, 1989.

[83] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77:257–286, 1989.

[84] K. V. Renaud. Guidelines for designing graphical authentication mechanism interfaces. *Int'l Journal of Information and Computer Security*, 3(1):60–85, 2009.

[85] K. V. Renaud. On user involvement in production of images used in visual authentication. *Journal of Visual Languages & Computing*, 20(1):1–15, 2009.

[86] B. S. Riggan, W. E. Snyder, X. Wang, and J Feng. A human factors study of graphical passwords using biometrics. *GCPR (in press)*, 2014.

[87] B. S. Riggan, W. E. Snyder, X. Wang, and L. K. Norris. A sketch-based authentication system with biometrics: security and usability analysis. *Pattern Recognition (Under Review)*, 2014.

[88] G. Rigoll, A. Kosmala, J. Rattland, and C. Neukirchen. A comparison between continuous and discrete density hidden markov models for cursive handwriting recognition. *Proc. of the 13th Int'l Conf. on Pattern Recognition*, pages 205–209, 1996.

[89] M. Rousson and D. Cremers. Efficient kernel density estimation of shape and intensity priors for level set segmentation. *Medical Image Comput. and Comp.-Ass. Interv. (MICCAI)*, 1:757–764, 2005.

[90] W. J. Rucklidge. Efficiently locating objects using the hausdorff distance. *Int'l Journal of Computer Vision*, 24(3):251–270, 1997.

[91] H. Sakoe and S. Chiba. Dynamic Programming Algorithm Optimization for Spoken Work Recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, ASSP-26(1):43–49, February 1978.

[92] B. Schölkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[93] B. Serra and M. Berthod. Subpixel Contour Matching Using Continous Dynamic Time Warping. *Proc. IEEE Computer Society Conference on Computer Visioan And Pattern Recognition*, pages 202–207, 1994.

[94] B. Serra and M. Berthod. Optimal Subpixel Matching of Contour Chains and Segments. *Proc. 5th International Conference on Computer Vision*, pages 402–407, 1995.

[95] B. W. Silverman. *Density estimation for statistics and data analysis.* Chapman and Hall, 1986.

[96] W. E. Snyder. A strategy for shape recognition. In A. Srivastava, editor, *Workshop on Challenges and Opportunities in Image Understanding*, College Park, MD, Jan. 2007.

[97] W. E. Snyder and H. Qi. *Machine vision.* Cambridge University Press, 2004.

[98] H. Tao and C. Adams. Pass-go: A proposal to improve the usability of graphical passwords. *Int'l Journal of Network Security*, 7(2):273–292, 2008.

[99] C. Varenhorst. Passdoodles: A lightweight authentication method. *MIT Research Science Institute*, July 2004.

[100] S. Weidenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Basic results. *Proc. of the 11th Int'l Conf. on Human-Computer Interaction*, 2005.

[101] S. Weidenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: Effects of tolerance and image choice. *Proc. of the 1st Symposium on Usable Privacy and Security*, pages 1–12, 2005.

[102] S. Weidenbeck, J. Waters, J. Birget, A. Brodskiy, and N. Memon. Passpoints: Design and longitudinal evaluation of a graphical password system. *Int'l Journal of Human Computer Studies*, 63(1–2):102–127, 2005.

[103] L. Yang, B. K. Widjaja, and R. Prasad. Application of Hidden Markov Models for Signature Verification. *Pattern Recognition*, 28(2):161–170, 1995.

[104] D. Yeung, H. Chang, Y. Xiong, S. George, R. Kashi, T. Matsumoto, and G. Rigoll. SVC2004: First International Signature Verification Competition. *Proc. of the Int'l Conf. on Biometric Authentication*, 2004.

[105] F. Yin, Q. Wang, and X. Zhang. Icdar 2013 chinese handwriting recognition competition. *Int'l Conf. on Document Analysis and Recognition*, pages 1464–1470, 2013.

[106] C. Zahn and R. Roskies. Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, 21(3):269–281, 1972.

[107] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37:1–19, 2004.

[108] H. Ziezold. On expected figures and strong law of large numbers for random elements in quasi-metric spaces. *Trans. of the 7th Prague Conf. on Information Theory, Statistical Decision Functions, Random Processes*, A:591–602, 1977.

APPENDICES

APPENDIX A

Optimization

In Chapter 5, the representative model was constructed by simply averaging the models. The average is optimal if the models being averaged are sufficiently close in the space of models and the space is approximately Euclidean. However, because the space is very high dimensional there are no guarantees, especially due to noise, hight intra-class variation, and discontinuities. Therefore, it may be useful to define an optimization problem.

Optimization is defined as "the science of determining the 'best' solutions to certain mathematically defined problems" [35]. In this chapter, the problem of constructing a representative model is posed as an optimization problem. Here, two approaches are suggested:

1. optimizing the squared error
2. optimizing the geodesic distance

Note that an optimal model, according to the objective function, does not necessarily mean that recognition performance will improves. The objective function must accurately

represent the problem. In many applications, the objective function being optimized is not the "real" problem, however, the "real" problem may be ill-posed and much too difficult to solve. Therefore, in some cases, a simpler objective function is constructed so that the solution may be well defined.

A.1 Squared Error

A common optimization problem involves minimizing the squared error, by some defined error term. This is widely used in neural network application [42]. The error in this application is considered the Euclidean distance between models. Directly, the optimization problem is stated as follows:

$$\underset{\varphi}{\text{minimize}} \quad \sum_{i=1}^{n} \left\| \varphi - \varphi_i \right\|^2$$

where φ and φ_i are models.

This, of course, yields

$$\varphi = \frac{1}{n} \sum_{i=1} n \varphi_i$$

which is the average model.

However, if we redefine the model as a function of the input sketch. This indirect optimization problem becomes:

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \sum_{i=1}^{n} \left\| \varphi(\boldsymbol{\alpha}) - \varphi(\boldsymbol{\alpha}_i) \right\|^2$$

It can be shown that the solution to this problem has the form

$$\sum_{i=1}^{n} \left(\varphi(\boldsymbol{\alpha}) - \varphi(\boldsymbol{\alpha}_i)\right) \frac{\partial \varphi}{\partial \boldsymbol{\alpha}} = 0.$$

Note the differences in these two optimization problems. The first seek to find the model φ that minimizes the squared error (in this case Euclidean distance) between each model, and as shown, the answer is the average model. The second problem seeks to find the sketch $\boldsymbol{\alpha}$ that minimizes the squared error between each model. Although, they are similar problems (and should result in a similar solution), the second problem relates the optimization back to the input sketch.

A.2 Geodesic

Similarly, if one considers to account for the nonlinearity of the "space of models" a geodesic distance may be defined, and then this distance may used in the optimization problem. Before we constructing this type of optimization problem, let us first define the term geodesic.

A parameterized curve γ is called a *geodesic* if the tangent vector field is parallel along γ [17]. Equivalently, if $D_\gamma \gamma' = 0$ (i.e. the acceleration vector is normal to the surface), where $D_\gamma \gamma'$ represents the covariant derivative of the vector field γ' with respect to γ.

Now, consider the following optimization problem:

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \sum_{i=1}^{n} \delta\left(\varphi(\boldsymbol{\alpha}), \varphi(\boldsymbol{\alpha}_i)\right)^2$$

where $\delta(\cdot)$ represents the geodesic distance between two models.

The solution takes the following form:

$$\sum_{i=1}^{n} \frac{\partial \delta}{\partial \varphi} \frac{\partial \varphi}{\partial \boldsymbol{\alpha}} = 0.$$

Note that derivatives may have to be approximated in order to obtain a solution.

APPENDIX B

Subspace Approximations

There are many methods, including PCA, RPCA, and KPCA, that attempt to reduced the dimensionality of models or descriptors. The advantage reducing the number of dimensions is to alleviate the effects from the "curse of dimensionality" [11]. This also produces a subspace approximation in which many times the models become more representative of their class.

In this chapter, both PCA and KPCA are discussed specifically in the context of the sketch-based password system introduced in this dissertation.

B.1 PCA

Given a set of observed feature vectors, $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, such that

$$\sum_{i=1}^{n} \mathbf{v}_i = 0,$$

PCA diagonalizes the covariance matrix:

$$C = \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i \mathbf{v}_i^T. \tag{B.1}$$

Note, that $\mathbf{v}_i$ is an alias for $\mathbf{v}_{\boldsymbol{\alpha}_d}^{(i)}$ from Chapter 6.

This decomposition is performed by solving the following eigenvalue problem

$$\lambda \mathbf{e} = C\mathbf{e}. \tag{B.2}$$

Let the set of eigenvectors $\left\{\mathbf{e}_1^*,\ \mathbf{e}_2^*,\ \ldots,\ \mathbf{e}_d^*\right\}$, which span the d-dimensional space, be the solution to the eigenvalue problem corresponding non-negative eigenvalues $\left\{\lambda_1^*,\ \lambda_2^*,\ \ldots,\ \lambda_d^*\right\}$ in descending order. Then, the principle components are consider to be the eigenvectors corresponding to the k largest eigenvalues, where $k < d$. These k principle components may be used to form a linear subspace, which is useful for reducing the dimensionality of the features $\mathbf{v}_i$ for $i = 1 \ldots n$.

Assuming the features $\mathbf{v}_i$ lie in d-dimensional space, each point may be projected into the linear subspace formed by the principle eigenvectors. That is,

$$\widetilde{\mathbf{v}}_i = \begin{bmatrix} \text{—} & \mathbf{e}_1^* & \text{—} \\ \text{—} & \mathbf{e}_2^* & \text{—} \\ & \vdots & \\ \text{—} & \mathbf{e}_k^* & \text{—} \end{bmatrix} \mathbf{v}_i$$

for $i = 1 \ldots n$.

In regard to this dissertation, a model (Equation 6.1) may be constructed from these projected features (with reduced number of dimensions). Therefore, the model equation

becomes the following:

$$m(\widetilde{\mathbf{v}}) = \frac{1}{z}\sum_{i=1}^{n} \exp\left(-\frac{1}{2}(\widetilde{\mathbf{v}}-\widetilde{\mathbf{v}}_i)^T\boldsymbol{\Sigma}^{-1}(\widetilde{\mathbf{v}}-\widetilde{\mathbf{v}}_i)\right). \tag{B.3}$$

Since dimensionality of the feature vectors effect the size of the model, reducing the dimensionality of the feature vectors makes computing the model more efficient. The danger in reducing the dimensionality too much is that the models from two distinct classes (i.e. different passwords) may be confused with one another.

B.2 KPCA

In the previous section, a subspace approximation of the model was constructed using PCA. Here, the subspace is constructed using nonlinear principle components, but the approach is very similar.

Consider a nonlinear mapping, $\Phi : \mathbb{R}^d \rightarrow \mathscr{G}$, where $\mathscr{G}$ denotes the feature space of arbitrary size. Similar to before, the observed features are assumed to be centered (this time in the feature space), meaning

$$\sum_{i=1}^{n} \Phi(\mathbf{v}_i) = 0.$$

Now, consider performing PCA (as discussed previously) in the feature space $\mathscr{G}$. Therefore, the covariance matrix

$$C' = \frac{1}{n}\sum_{i=1}^{n} \Phi(\mathbf{v}_i)\Phi(\mathbf{v}_i)^T$$

is diagonalized by solving the following eigenvalue problem:

$$\lambda\mathbf{E} = C'\mathbf{E} \tag{B.4}$$

This time the eigenvectors, $\left\{\mathbf{E}_1^*,\ \mathbf{E}_2^*,\ \ldots,\ \mathbf{E}_d^*\right\}$, lie in $\mathcal{G}$ instead of $\mathbb{R}^d$. Therefore, when the eigenvectors are projected back into the original space, they are nonlinear. Using the k principal eigenvectors, a similar approach may be used to project the feature vectors into the nonlinear subspace constructed using KPCA and then construct a model.

There are, however, some considerations when using KPCA. One consideration is the fact that the mapping, Φ, is unknown. Therefore, the eigenvalue problem is solved using kernel methods; hence, "kernel" PCA. Another consideration is the centering the data in the input space does not guarantee that the projected data is also centered. Therefore, a slight modification must be made (see [92]).