Handwritten signature authentication using motion detection and QRCodes

Ricardo Manuel Pereira Gonçalves
Mestrado em Ciência de Computadores
Departamento de Ciência de Computadores
2015

Orientador
Prof. Manuel Eduardo Carvalho Duarte Correia, Professor Auxiliar, FCUP

Coorientador
Alexandre Barbosa Augusto, Investigador, CRACS/INESC

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, ______/______/_________

To my family, to my friends and to my love for all their patience and unconditional support that were essential to make all this possible...

Acknowledgements

I have to thank my advisor professor Manuel Eduardo Correia for all his unconditional support and guidance that made this work possible. I also want to specially thank Alexandre Augusto for all his support and useful tips during the development of this work.

Lastly I want to thank FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project UID/EEA/50014/2013 in conjunction with institutional support provided by DCC/FCUP and the facilities and research environment gracefully provided by the CRACS (Center for Research in Advanced Computing Systems) research unit, an INESC TEC associate of the Faculty of Science, University of Porto.

Resumo

O reconhecimento de assinaturas manuscritas é dos método mais aceites para validar documentos em papel. No entanto, no mundo digital, é difícil distinguir um documento digitalizado com uma assinatura manuscrita real de um documento falsificado. Um documento pode ser forjado com uma cópia da assinatura feita pela mesma pessoa noutro documento e simplesmente "colada" no documento forjado. Os documentos digitais podem ser protegidos com assinaturas e certificados digitais, mas nem sempre são bem aceites pelo público geral. Um documento com uma assinatura manuscrita pode até ser melhor aceite do que um documento certificado digitalmente. Infelizmente, isso impõe restrições no uso de documentos digitais, obrigando a que os documentos assinados continuem a ser baseados em papel, tornando todo o processo mais caro e lento.

Com este trabalho pretendemos apresentar uma solução baseada em biometrias que permita juntar os dois mundos. Por um lado, queremos oferecer a segurança de um documento digital e por outro lado, queremos dar a "garantia" de um documento assinado à mão. O sistema que vamos propor faz uso das telas sensíveis ao toque dos smartphones ou tablets para adquirir as imagens da assinatura manuscrita e dos marcadores biométricos. Estes marcadores são derivados não só da direcção dos movimentos feitos diretamente no dispositivo com uma caneta omnidirecional ou com o próprio dedo, mas também dos movimentos de tocar e levantar que o utilizador faz. Os dados obtidos são então comparados com os marcadores biométricos do utilizador previamente registados. Neste trabalho sustentamos que a recolha destes dados biométricos simples permitem que se faça a verificação automática de assinaturas manuscritas em documentos digitais e autenticação de utilizadores em serviços online. Para tornar o sistema mais interativo e fácil de se utilizar, iremos fazer uso de QRCodes devido à sua comodidade de utilização.

Abstract

Handwritten signature recognition is still the most widely accepted method to validate paper based documents. However, in the digital world, is hard to distinguish a scanned document with real handwritten signature from a forged document. A document can be forged with a copy of the signature made by the same person on another document and simply "pasted" into the forged one. Digital documents can be protected with digital signatures and certificates, but they are not always well accepted by general public. A document with a handwritten signature can be more accepted than a document digitally certified. Unfortunately, this imposes restrictions in the use of digital documents, requiring that signed documents continue to be paper-based, making all process more expensive and slower.

With this work we want to present a biometric-based solution that will join the two worlds. On one hand, we want to be able to offer the security of a digital document and on the other hand, we want to give the "guarantee" of a document signed by hand. We will propose a system that make use of the touchscreens of smartphones or tablets to collect handwritten signature images and biometric markers. These biometric markers are derived not only from direction of motion that is made directly into the device with a omnidirectional tip stylus or with the finger, but also the dragging or lifting movements the user makes. The obtained data is then compared with previously enrolled biometric markers of the user's handwritten signature. In this work we contend that collecting these simple biometric features allow to make automatic verification of handwritten signatures in digital documents and user authentication in online services. To make the system more interactive and easy to use, we will make use of QRCodes due to its convenience of use.

Palavras-chave/Keywords

Palavras-chave:

- Assinatura manuscrita
- Biometrias
- Verificação de assinaturas
- Autenticação
- Dispositivos móveis
- Distância Euclidiana

Keywords:

- Handwritten signature
- Biometrics
- Signature verification
- Authentication
- QR Code
- Mobile devices
- Euclidean distance

Acronyms

CA	Certificate Authority. 21
DTW	Dynamic Time Warping. 14, 17
EER	Equal Error Rate. 12, 16, 17, 55, 56
FAR	False Acceptance Rate. 11, 12, 15, 16, 55
FHE	Forensic Handwriting Examiners. 17
FMR	False Match Rate. 11
FNMR	False Non-match Rate. 11
FRR	False Rejection Rate. 11, 12, 15, 16, 55
HMM	Hidden Markov Models. 15
HTML	HyperText Markup Language. 23, 27, 28, 50
HTTP	Hypertext Transfer Protocol. 23, 26, 27, 50
HTTPS	HTTP Secure. 23
JDK	Java Development Kit. 68
JSON	JavaScript Object Notation. 50, 51, 68
JSP	JavaServer Pages. 68
MITM	man-in-the-middle. 23
PDF	Portable Document Format. 20, 22

PKI	Public Key Infrastructure. 22
QoS	Quality of Service. 27
QRCode	Quick Response Code. v, vi, xi, 3–5, 19, 25, 26, 30, 65, 69, 72, 78
REST	REpresentational State Transfer. 26, 27, 50
SMTP	Simple Mail Transfer Protocol. 26
SOA	Service-Oriented Architecture. 26
SOAP	Simple Object Access Protocol. 26, 27, 50
SSL	Secure Sockets Layer. 23
SVM	Support Vector Machine. 15, 18
TLS	Transport Layer Security. 23
URI	Uniform Resource Identifier. 27
URL	Uniform Resource Locator. 26, 30, 72, 78
XML	Extensible Markup Language. 26, 27

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Many modern society processes are rooted on written notarized contractual documents that then act as proof of certain actions or facts about individuals or other entities that result from many societal interactions like, for example, enterprises or persons establishing legal binding contracts between them. These paper based documents must then be laden with security artifacts that make them hard to forge and attest about the document authenticity, integrity and the time it was produced. They often act as root proof of important facts, e.g. to proof who is the owner of a property, if some payment was made and on what circumstances, prove what the monthly income is in an employee contract, etc. Over the years society evolved some overall well-accepted procedures and security artifacts that allow one to consider these documents as authentic and acceptable in a court of law, the most common of which is the handwritten signature and the embossing institutional stamp.

Unfortunately these highly accepted societal procedures base their security on the intrinsic physical properties of the paper that is used to print and hand sign or stamp these documents. For digital documents, these physical properties of paper do not hold anymore and therefore we need to adopt other security mechanisms if one wants to be able to produce digital documents with security properties analog to what can be cheaply and easily achieved with paper. In the real world, due to the physical characteristics of paper, handwritten signatures and embossing stamps constitute privileged ways to secure and attest the integrity and authenticity of printed documents. However in the digital world, is hard to distinguish real handwritten signature on a scanned document from a forged copy of another signature made by the same person on another document that is simply “pasted” into the forged document. As soon as one paper document with a real handwritten signature is scanned, anyone who can access a copy can then use the scanned handwritten signature to impersonate the signature owner in every other kinds of digital documents, by simply pasting an image of the handwritten signature into the forged document. However, some of these forgeries can be detected using copy-move forgery detection techniques [6]. But, even so, we can only safely attest on the authenticity of a handwritten signature that is made on the original physical paper document. This is rather unfortunate because this implies a waste of paper resources and the involvement of a trusted third party (a notary service) each time a copy of the original document is needed. This process is obviously quite expensive and highly inefficient.

The standard accepted method of authenticating digital documents is using digital signatures based on X.509 digital certificates. This is a well-accepted legal method to secure digital documents. However, due to its technical complexities, certificate based digital signatures are still not well accepted by the general population. One as only to remember that for example a great percentage of the Portuguese population already has an identification document ("cartão do cidadão"), capable of performing legal binding digital signatures, but that almost no one uses. Culturally, at least in the western world, the general public still has more affinity and can more easily relate security with a visual analysis of handwritten signatures. This has been true historically and is still valid today [13].

With this work we want to start building a better bridge of confidence between the general public and digital documents whose security is based on the unique characteristic of biometric markers that can be derived from the dynamics of movement of the human hand when it performs a handwritten signature, something everyone on a western culture is trained to do. From these we can then derive security digital artifacts to include into digital documents, to attest their authenticity. We strong believe that security mechanisms for digital documents, based on what people are already used and trained to do, to secure paper based documents, have more chance of success and general acceptance then the more classical, but less well accepted approach of securing digital documents with the technically more challenging certificate based digital signature mechanisms.

1.1 Motivation

Today we live in a more and more digital world. With that, the society have to adapt to the new challenges that come from it, this means that new procedures have to be adopted that enables the same procedures that were used before or something equivalent. This is what happens with signed paper documents, we have to find out new ways to have the same authenticity in digital documents.

With advent of new authentication mechanisms that we assist today, like the lock pattern and the integration of fingerprint scanner in smartphones, we also want to explore new ways to provide authentication. As we will discuss later, there are many ways that we can benefit from an automated system capable of recognizing handwritten signatures.

This is an important and actual research topic, with new achievements everyday. Therefore, this work is suitable in the context of the society the we live today.

1.2 Proposed Solution

With this work we pretend to build a system to be used when someone needs to be authenticated. The authentication will be made using the user's handwritten signature, like is done with paper based documents. We also pretend make use of smartphones because they are ubiquitous nowadays. To accomplish this we want create a web service which will allow the users to register their signatures to be used later for authentication. The system will have three components:

1. a *web service (Authenticity Service)* which will provide the service
2. a *client* which will be the interface between the web service and the user
3. a *mobile device* which will be used in interaction with the client through QR-Codes to acquire the signatures

The Authenticity Service will be responsible to verify authenticity of a user. It will act like the notary service that we have in our society. When someone have the need for an authentic document he have to go to a notary to ask for that. The notary is a trusted and authorized entity in which everybody rely on. For that reason, the Authenticity Service must be also trusted and cannot be compromised.

When someone wants to use the system should enroll first. During that process, the user can be asked to sign several times to practice the system. Those signatures will be used to create his biometric features.

When is necessary to verify the authenticity, the user, with a smartphone, should read a QRCode, open the application referenced by it and then sign and submit his signature. The Authenticity Service will reply if the signature is authentic.

With this system is possible to give some degree of confidence of authenticity to a digital document. To improve reliability it can be used with another authentication system.

1.3 Objectives

The main goals of this work is to define and implement a biometric system based on handwritten signatures by the means of mobile devices. We identified the following main objectives:

- **Research** of the current state of the art on handwritten biometric authentication systems
- **Definition** of an architecture that creates an authentication mechanism based on handwritten signatures
- **Implementation** of the proposed architecture
- **Deployment** of the system in real scenarios to analysis the performance and usage
- **Proof of concept** that with mobile devices is possible to recognize users and provide some authenticity based on biometric handwritten signatures. Despite they are not specialized devices to this application (they are not capable of acquiring for example pressure and pen inclination), they are ubiquitous and they have a huge potential
- **Dissemination** the results with its stakeholders and encourage them to use this system as a means of authentication

1.4 Features

The system should be able to fulfill the following features:

- **Easy to use** - the system should be easy to use, otherwise the users will not feel comfortable using it and leading to unsuccess
- **Reliability** - the system should be trusted and secure, otherwise it can be compromised and provide incorrect authenticity
- **Provide authentication** - the system should be capable of recognizing the signatures of the users with high degree of accuracy
- **Highly available** - the system should be accessible to be used by a large number of users
- **Quick to respond** - the system should be fast providing results, i. e., it should have high performance

1.5 Application Domain

This system has a very wide range of applications in several domains. It can be used from electronic e-commerce, passing through e-government up to the health sector, it can have implications in almost every situation where handwritten signatures are used.

The initial application was the ability to provide some authenticity to handwritten signed digital documents. The user places his handwritten signature inside a document with a QRCode associated. Later, is possible to verify the authenticity of the signature with that information (we will describe this process in more detail later).

Another possible use case is the replacement of specialized devices that are used in express delivery services. When we receive registered mail, we have to make a signature to, in some way, prove that the mail was received. By using this application, not only will help to reduce the costs, because an ordinary smartphone is sufficient, but also will provide some authenticity if is necessary to verify the signature. This can replicated for other similar services.

It can have even applications in cases that are not used today, for example, it could replace the password authentication in websites by handwritten signatures or be used as two-factor password. This will allow to improve the security when accessing to the online banking account, accessing to governmental services or even accessing to email account. Many of online services that require authentication can benefit from a system like this.

The applications of this system are almost endless, almost every system which requires authentication can take advantage.

1.6 Contributions

During the project development, a paper was published in 10th Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI 2015), titled "Time/space based biometric handwritten signature verification" [21] held in Águeda, Portugal.

1.7 Research methods

The literature review we have conducted to better contextualize the research and work we have done for this thesis was performed using the IEEE Xplore, Google Scholar and Science Direct search engines. We have applied the following queries:

- **IEEE Xplore** - "`[ANY FIELD] (((handwritten OR online) AND signature) AND (verification OR recognition)) OR biometrics`"
- **Google Scholar** - "`[ANY FIELD] handwritten signature OR online signature verification OR biometrics`"
- **Science Direct** - "`[ANY FIELD] (((handwritten OR online) AND signature) AND (verification OR recognition)) OR biometrics`"

Due to the high number of results thus obtained we had them filtered by their abstracts, according to the following inclusion criteria:

1. English as language
2. Recent articles
3. Articles with high relevance on the search engines
4. Well known and highly referenced authors in the field of digital identity management

5. A cautiously review of titles, abstracts, keywords and section titles, to exclude papers with the same or somewhat outdated subjects

After the analyzing the most promising papers/standards texts, their citations were also reviewed and those that suited the inclusion criteria were also integrated in the review. The search and full text retrieval of the selected papers was performed in the following databases:

- Biblioteca do Conhecimento Online (b-on)
- Open Repository at the University of Porto
- Open Access Scientific Repository of Portugal (RCAAP)
- Google Scholar

In the end, we have selected eight papers which were selected by keywords [24, 52, 22, 12, 25, 9, 47, 13].

1.8 Outline

Here you can find a small summary describing what will be discussed in the next chapters:

- **Chapter 2 State of the Art** - will be discussed about the research done for this work, what is the current status in this field of study and current applications. We will also introduce important basic concepts necessary for this work.
- **Chapter 3 Implementation** - will be described the details about general architecture and the actual implementation of the system
- **Chapter 4 Testing and Results** - we will talk about the current achievements and improvements made in the system
- **Chapter 5 Conclusions and Future Work** - discussion about the main findings of this work, current limitations and the future directions

Chapter 2

State of the Art

Handwritten signatures are one of the fastest growing global practices because of their convenience [13]. In this chapter we will review the state of the art in the field of study of the this work. We will start introducing base concepts to understand the subject we are addressing and the strategies to solve it. Then we will discuss how to optimize the system and the current performances achieved by other projects. Further, we will talk about more technical aspects, like security, mobile devices technicalities and communication concepts. Finally we show some similar applications existing in the literature.

2.1 Handwritten signature

A signature is a handwritten representation of the author's name or nickname or any other mark. Each person has his own way of doing signatures and replication is almost impossible. With forensic analysis it is possible to determine if a signature is authentic or forged with high accuracy and even with legal value [18]. Figure 2.1 shows some examples of handwritten signatures.

From around 5$^{\text{th}}$ to 3$^{\text{rd}}$ millenium BC, the Sumerian civilization began to develop writing. Along with that they created methods to authenticate the author of the writings like seals and complex works of art. Only during the Roman Empire became common practice use writing as a form to authenticate documents. Since then handwritten signature gained the importance that we know today, it is one of the main forms to authenticate paper documents [2].

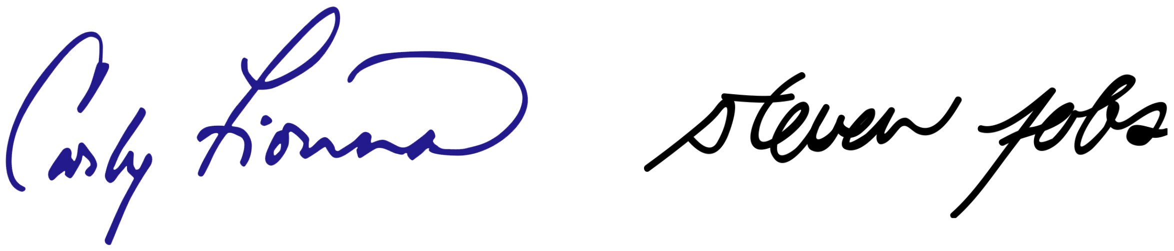


Figure 2.1: Examples of handwritten signatures

With the appearance of electronic devices and the development of new areas, like signal processing, led to the emergence of new ways to identify people and a increased interest of governments to automate processes. It was in the 1980s that appeared the first electronic biometric systems based on handwritten signatures [57].

During the 1940s, the introduction of ballpoint pens forced the forensics specialists to adapt and find new methods to analyze signatures. Something similar happens today with electronic signature, it is another paradigm shift. New methods and tools should be developed to face the new challenges [18]. In fact, it is a very active field of study and new solutions arise every day.

However, not much is known what people perceive about digital handwritten signatures to be symbolically equivalent to traditional hand signatures. Some studies demonstrate that although functionally the same, digital handwritten signatures evoked markedly different psychological reactions than signatures written on paper. Namely, digital handwritten signatures evoked a weaker sense of the signer's presence and involvement. This weaker sense of social presence, in turn, induced negativity: people were more likely to discount the validity of an e-signed application than a paper signed document [13].

2.2 Biometrics

Biometrics or biometic recognition is a field of study that uses distinctive physiologic (like fingerprinting) and/or behavioral (like calligraphy) characteristics and identification techniques (like statistical methods) to identify automatically a person [36].

According to James Wayman et al [57], they define:

— "Biometric technologies" are automated methods of verifying or recognizing the identity of a living person based on a physiological or behavioral characteristic —

There are many examples of biometric systems that are used nowadays. They can use one measurement or many measurements to improve accuracy.

The first biometric systems were introduced by Alphonse Bertillon in the 1870s to identify prisoners based on body measurements such as skull diameter and arm and foot length. Only from the 1960s that biometrics systems began to have a more substantial development. It began with the emergence of speaker and fingerprint recognition and then with hand geometry systems. Later appeared the retina and signature verification followed by face systems and more recently based on iris recognition [57].

But there many more examples of biometrics that can be used, for example gait analysis, keystroke dynamics, vein matching, skin tone and voice analysis. Each method have its characteristics and some of them are more accurate than others. For example, fingerprint recognition is less accurate than iris recognition, but on the other hand, a system that uses fingerprint recognition is more affordable than a system that uses iris recognition.

2.2.1 Authentication and identification

We can use biometrics to authenticate a user in a system (1-to-1), i. e. comparison. The user that wants to be authenticated has to provide a fresh new biometric data from himself. He will be authenticated if the biometry he provided matches with the one registered in the system. [42].

Biometrics can also be used for identification (1-to-many). The user provides his biometrics and the system must be able to find out who is the owner. If a match occurs, that will identify who is the user assigned with the biometry. To make this process feasible for a large number of users, the system have to implement special techniques to make it efficient. In recent years, biometrics have been used mainly for identification instead authentication. Identification systems based on biometrics should take special caution with the privacy of biometric data [42].

2.2.2 Biometric system properties

For the same biometry, different systems will perform differently by giving better or worse results. A biometric system should have the following qualitative characteristics to lead to good results [57, 56]:

- **Universality** - each potential user possesses the biometry
- **Uniqueness** - the biometry adequately differentiates between any two users
- **Permanence** - the biometry profile remains relatively constant over time
- **Distinctiveness** - the population should show enough variation among individuals
- **Availability** - if it is possible to measure the biometric features multiple times
- **Collectability** - the biometric samples are easy to detect and acquire
- **Acceptability** - if users are generally willing to accept and use the biometry

These enumerated qualities can be quantified with many rates. We will focus in the main ones:

- **False Rejection Rate (FRR) or False Non-match Rate (FNMR)** is the probability of a user with a true identity be rejected
- **False Acceptance Rate (FAR) or False Match Rate (FMR)** is the probability of a user with a false identity be accepted, thus allowing fraud
- **System Throughput Rate** is the number of users that the system can handle in a time period
- **User Acceptance** will vary depending on how it will be accessible to the public, how is user-friendly and how is marketed
- **Cost savings** made with the implementation of the system

Contrary what could be expected, it is very difficult to compare two biometric systems. All biometric systems are highly dependent on how they are implemented, on population and in hardware/software used. Nevertheless, biometric systems administrator have some freedom to tweak the system to make it perform better by improving the parameters used [57].

The system will become better as much as how possible to keep FRR, FAR and costs low and System Throughput Rate and User Acceptance high. But to find the right balance between these parameters can be a surprisingly difficult task [58].

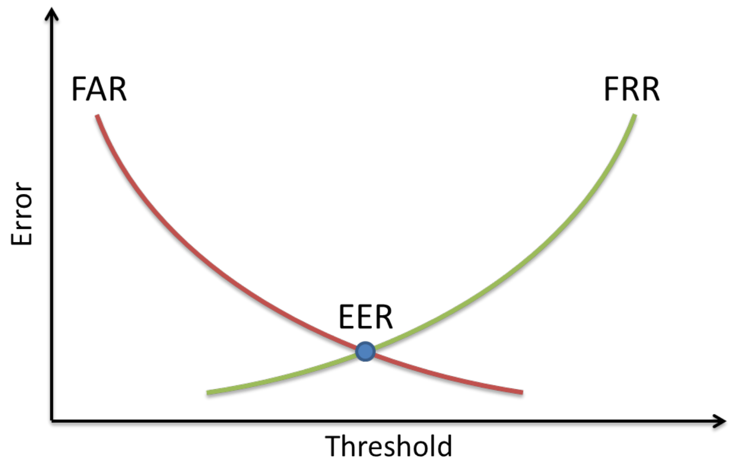


Figure 2.2: Representation of a biometric system performance

Another parameter that can be used is the Equal Error Rate (EER) which allow to quickly understand the performance of the biometric system. EER is the value where FAR and FRR are equal, i.e., where $FAR = FRR$. In general, systems with lower EER are better.

Figure 2.2 relates the threshold with the error rates and shows the general behavior of FRR and FAR. With higher threshold the system is prone to accept everything, so FRR is low and FAR is high. If we decrease the threshold, FRR will increase and FAR will decrease. But if the threshold is too low, the system become unusable because is very hard to get accepted. EER is usually the best balance, but we can have some cases where we should provide a different kind of answers. If we want to provide results that have high degree of confidence, we should be very restrictive with FAR and, consequently, it becomes harder for a user to get accepted because the FRR is high.

2.3 Signature verification

Signature verification is not a trivial problem to solve, completely guarantee the authenticity of a signature is nearly impossible. Several approaches and methods can be used to tackle this problem, some of them provide better results than others. Next, we will introduce the types of verification that can be made, the process of verification of a signature and, finally, the main methods used to verify a signature with another.

2.3.1 Types of verification

There are two types of verification that can be made, offline or online verification [25]:

- **Offline or static** - in offline verification, the signature is made on paper and then scanned. The result is a bitmap image and all process of verification should be rely on it. The methods used in this kind of verification can be very different in comparison with online verification. Since the image is scanned, all information regarding to dynamics and time is lost. Even for segmentation, the approach should be different.
- **Online or dynamic** - in online verification, the signature is made directly in the digitizer device. There are specialized devices for this purpose, but general purpose devices like smartphones and tablets are also able to acquire the signature. Specialized devices are better because they have better resolutions and the result has more precision.

This work will focus on online verification because it offers a more reliable authentication and higher security level than offline signature verification due it is based on the dynamics of the signature of a person [22].

2.3.2 Verification process

The verification process is divided in several phases: data acquisition and preprocessing, feature extraction and classification [25]. Next is a brief explanation of each one:

- **Data acquisition and preprocessing** - first of all, the data should be acquired from the user using a device. Then, the data should be preprocessed in order to be stored.
- **Feature extraction** - in this phase, all features used on comparison are extracted from the signature, for example, if the method is based in acceleration, then acceleration should be extracted.
- **Classification** - after the features are extracted, now is possible to compare the signatures and produce a result, saying if the signatures match or not.

2.3.3 Data acquisition

The first step in handwritten signature verification is acquire the signature. In the market there are several devices capable of acquiring the signature. They can differ from each other allowing to acquire different parameters, changing the accuracy of the signature verification.

Mobile devices have been sold all around the world and they can be used to acquire signatures. Typically, they are able to acquire position, time, when the touch starts and when it ends. But there are specialized devices which are capable of acquiring other parameters, like pressure, force, direction of movement and pen inclination [25].

Despite mobile devices are a good option because they are widespread, they can have some restrictions. They can have a small input area and poor sampling frequency, making the signature not very accurate [24].

2.3.4 Verification methods

To verify a handwritten signature is not easy and can be a complex task. The main goal is to guarantee that a forger cannot authenticate the signature. Of course, it is impossible to completely guarantee a forger is not able to authenticate a signature because if he knows exactly how the original signature is, he will be able to forge it. So, the forger's task should be as hard as possible.

There are almost endless methods to verify a signature. We only discuss about the main approaches commonly used:

- **Euclidean distance** - this method is based on distances measurements. The distance between the test signature and the reference signature should not exceed some threshold [25, 29].
- **Correlation** - measures the relation between two signatures. The result is a score in range [-1, 1]. When the result is closer to 1 means that signatures are similar, closer to 0 means that the signatures are different and closer to -1 means that the signatures are the opposite of each other. The signatures can be divided in regions and then correlate each region. This is called regional correlation [43].
- **Dynamic Time Warping (DTW)** - this method measures the difference between two temporal series. Those temporal series can have different speeds

(consecutively different accelerations), but the similarities can be detected in walking patterns. This algorithm was originally used in speech recognition [40].

- **Support Vector Machine (SVM)** - this method is a kind of learning machine for pattern recognition and regression problems, which constructs its solution in terms of a subset of the training data, the Support Vector. The method is popular in various pattern recognition problems because it provides very good results [17].

- **Neural networks** - this method is a statistical method used in machine learning and is inspired by biological neural networks (which is the case of brain) and are used to estimate or approximate functions with large inputs and those functions are generally unknown. Neural networks usually are systems of interconnected neurons which can compute values from the input [25, 28, 38]. An example of its use in handwriting recognition is neurons being activated by the pixels of an input image. Then the network is weighted and transformed by a function and the activations are passed to other neurons. This process is repeated until the output neuron is activated. Examples of neural networks are Bayesian, time-delay and back propagation networks [25, 28, 38].

- **Hidden Markov Models (HMM)** - this method is a doubly stochastic process governed by an underlying Markov chain with a finite number of states and a set of random functions each of which is associated with one state. At discrete instants of time, the process is in one of the states and generates an observation symbol according to the random function corresponding to the current state. The model is hidden in the sense that all that can be seen is a sequence of observations. The underlying state which generated each symbol is hidden [61, 27, 23].

Each method is not restricted to a feature or a set of features. Who is implementing the system should select the collection of features that are more suitable for his system. But more features used, more likely the system become better. Each approach (method and set of features) will lead to different ways to train the system.

2.4 System optimization

Biometric systems usually are configurable to be more restrictive or more permissive by changing the FAR and FRR rates. It is possible adapt the system to have $FAR =$

FRR, i.e. the EER. This is the point where the system will work better, but, in some cases, we want to be very restrictive with FAR like in critical systems.

To find the best EER we can use an approximation method which successively iterates and each time gives better solutions until is not possible to improve more. The method is based on an extension of the simplex algorithm for non-linear programming, which makes it suitable for both unconstrained and constrained optimization. With several extensions of the method we can escape local optima making it a good tool for optimization of nonlinear functions with many local optima. A strategy which proved to be extremely robust was random start local search with a correct setup. The idea is to use a very large simplex at the begin; the initial movements of this simplex are very large, and therefore act as a kind of filter, which naturally drives the search into good areas [45]. So, as objective function, we can choose to minimize the EER.

To measure the effectiveness of the biometric system we have to test it with a populated database with both authentic and forged signatures. The introduction of forged signatures in the system is essential to understand what is the possibility of an intruder have unauthorized access. The forgeries can be classified as [48]:

- **Random forgery** - the forger does not have access to the signatures nor the author's name, the forger makes a random signature
- **Simple forgery** - the forger have access to the author's name but not to the signature, the forger makes the signature on his style
- **Skilled forgery** - the forger have access to the author's name and a sample of the signature, the forger is able to reproduce the signature

The major task of any signature verification system is to detect whether the signature is genuine or not. A signature forgery means an attempt to copy someone else's signature and use them against him to steal his identity [11]. The forgery harder to detect is naturally the skilled forgery, but the systems must be prepared to avoid that.

2.5 Comparison of signature verification systems

There are competitions organized where participants put their algorithms subjected to a series of tests. Each competition has its own rules, datasets and ways to evaluate the results.

The measurements commonly used in the competitions to compare algorithms are the traditional EER and forensically substantial Cost of Log Likelihood Ratios $\widehat{C}_{llr}$. The likelihood ratio is a comparison score (e.g. a degree of similarity or difference), and the evidential value of that score, expressed as the ratio of the probabilities of finding that score when the questioned signature is a genuine signature and when it is a forgery. It is defined, according with Forensic Handwriting Examiners (FHE), as the ratio between two hypotheses (at least) of the observations ("Hipothesis Testing" formulation) [35]:

- H1: The questioned signature is an authentic signature of the reference writer
- H2: The questioned signature is written by a writer other than the reference writer

The true issue is to find the likelihood ratio for a comparison: the probability of finding a particular score given that Hypothesis H1 is true, divided by the probability of finding the score when the alternative Hypothesis H2 is true. H1 corresponds to intra-source scores (same author) and H2 to inter-source scores (different authors) [35]. Systems with smaller values for EER or for Cost of Log Likelihood Ratio are better [35].

The context where the system is used it can cause huge implications in the performance, for example different typewriting styles. Following, we will show some results from recent signature competitions. We will focus in the recent competitions SigWiComp 2013 and SigComp 2011:

1. **SigWiComp 2013** [35] - the tests made were for online signature verification of Japanese handwriting, as shown on table 2.1:

ID	Institution	Method	Accur.%	FRR	FAR	$\widehat{C}_{llr}$	$\widehat{C}_{llr}^{min}$
11	Qatar University	Differences between features	70.55	29.56	30.22	1.176164	0.884223
12	Sabanci University	DTW	72.55	27.56	27.36	1.089183	0.744544
13	Sabanci University	DTW with pressure	72.47	27.56	27.5	1.114015	0.744793

Table 2.1: Results of SigWiComp 2013 for Japanese signature verification

2. **SigComp 2011** [33] - the tests made were for online signature verification of Chinese and Dutch handwriting, as shown on table 2.2 and on table 2.3 respectively:

ID	Institution	Method	Accur.%	FRR	FAR	$\widehat{C}_{llr}$	$\widehat{C}_{llr}^{min}$
1	Sabanci University	Tesselation and SVM	84.81	12	16.05	0.565146	0.351142
4	xyzmo (1)	Statistical	93.17	6.4	6.94	0.413413	0.217915
5	xyzmo (2)	Statistical	93.17	6.4	6.94	0.418631	0.217915
6	Qatar University	Probability distribution	82.94	16.8	17.14	1.049099	0.503151
7	Qatar University	Probability distribution	85.32	13.6	14.97	0.905516	0.46114
9	Anonymous-2	—	80.89	9.26	8.14	6.210251	0.733883

Table 2.2: Results of SigComp 2011 for Chinese signature verification

ID	Institution	Method	Accur.%	FRR	FAR	$\widehat{C}_{llr}$	$\widehat{C}_{llr}^{min}$
1	Sabanci University	Tesselation and SVM	93.49	7.56	7.69	0.492844	0.23755
4	xyzmo (1)	Statistical	96.27	3.7	3.76	0.258932	0.122596
5	xyzmo (2)	Statistical	96.35	3.86	3.44	0.351189	0.122596
6	Qatar University	Probability distribution	91.82	8.33	8.02	0.534542	0.29094
7	Qatar University	Probability distribution	92.93	7.25	6.87	0.604641	0.241201
9	Anonymous-2	—	88.56	11.11	11.27	6.433622	0.408429

Table 2.3: Results of SigComp 2011 for Dutch signature verification

These results serve to get a general sense about the state of the art of handwritten signatures recognition and which are the performances achievable. The results of SigComp 2013 are different of SigComp 2011 due to different datasets and also different parameters. Other competitions which may also be interesting to analyze are: 4NsigComp 2012 [32], 4NSigComp 2010 [31] and SigComp 2009 [7].

2.6 Similar applications

After reviewing the literature, still does not exits many applications that combines documents authentication with handwritten signatures and QRCodes. The system that we are proposing is, in some way, related with touch dynamics, which is an active emerging research area. Touch dynamics is the recognition of user's authenticity using the touches made by him in the mobile device, for example the touches made in the *lock-pattern*. With this, it is possible to recognize the user's authenticity with high accuracy [5, 39].

Next, we will show a brief description of two systems that combine QRCodes and handwritten signatures and then two commercial solutions, Docu*Sign* and SIGN*ificant*, that use handwritten signatures to authenticate documents.

2.6.1 Handwritten signatures and QRCodes

Puchong Subpratatsavee et al [52] combine QRCodes and handwritten signature verification to verify the ownership of a card to be used in the authentication of financial transactions and contracts with government. Initially, user signs the original signature on the smartphone. Then, the signaling waves of the signature are converted to binary format. The binary data is encrypted with public key of the card issuer and generate a QRCode for printing it into the back of the ID card instead of using signatures. When the user needs to confirm the deal will be required a signature, the card's issuer can scan the QRCode. Who claim to be the owner of the card must sign his own signature on the smartphone and the system decrypts the binary data on the QR Code with the issuer's private key to obtain the signature of the card holder. Finally, the system compares the two signatures with an algorithm that combine motion and direction. Handwritten signatures in association with QRCodes and encryption with public key system can prevent a claim from thieve with a stolen ID card to copy signature with high accuracy.

Another resembling system is the one proposed by Anu Chaudhary et al [11]. When a user registers, the signature is processed and the features are extracted and printed on a QRCode on bank cheque. To verify the authenticity, a new signature is gathered from the user and compared with information on the QRCode. The QRCode contains the features of the signature which can be used as a PIN.

2.6.2 Docu*Sign*

Docu*Sign* (`https://www.docusign.com`) is a platform targeted for signing documents. The user starts the process by uploading the documents that want to sign to the platform and specifying the names and email addresses of the signers. It is also allowed to insert tags to indicate where a signature should be placed, initials, dates or add custom data fields for signers to fill in. Then the documents are sent to recipients and they will be able to access them. Once the document is complete, it is stored securely for easy retrieval [15].

The platform is responsible for managing the authenticity and security of the documents. None signature verification is made during the process, but the platform ensures secure electronic signatures and legal acceptance. Some PDF readers have this system integrated in the application, like Foxit Reader [51]. Meanwhile, other applications provide very similar systems, like Adobe Acrobat Reader which integrates its proprietary service Adobe EchoSign [14].

2.6.3 SIGN*ificant* by XYZMO

The SIGNificant platform (`https://www.xyzmo.com/`) provides a user interface and tools needed to define an e-signature process. It is possible to use signature pads, interactive pen displays, mobile devices or Web-based signing. The platform's building blocks allows to choose a combination of capturing devices for each use case.

The handwritten signature is recorded using all possible parameters [60], such as acceleration, speed, and rhythm. This makes the digitized signature forensically identifiable. If necessary is possible to use an expert tool that forensically analyzes the biometric characteristics of the captured signature. It is also provided a signature verification that authenticates a signature against a pre-enrolled signature profile database, in real-time.

The signature data is encrypted asymmetrically using a pair of private/public key while the signature is being record [60], making a signature securely bounded to a unique document. To prevent sniffing of the data, it is also offered security mechanisms for encrypting the communication between the signature pad and the computer and the encryption of the data on the pad itself [60].

2.7 Security concepts

It is remarkably easy to gain unauthorized access to information in an insecure networked environment, and it is hard to catch intruders. Seemingly innocuous information can expose a computer system to compromise. Information that intruders find useful includes which hardware and software are being used, system configuration, type of network connections, phone numbers, and access and authentication procedures. Security-related information can enable unauthorized individuals to access important files and programs, thus compromising the security of the system. Examples of important information are passwords, access control files and keys, personnel information, and encryption algorithms [46]. Biometric information is also sensible information that must be protected from intruders, especially when used as a mean of authentication and identification.

Next, we will discuss about X.509 certificates that are used to secure documents and communications over the network.

2.7.1 X.509 digital certificates

Digital certificates allow an entity to trust in another. Different entities (like users, institutions, processes or devices) make use of digital certificate infrastructures to secure their communications and data, ensuring confidentiality, integrity, authenticity and non-repudiation. These caracteristics can be achieved using ciphers, digital signatures and authentication protocols. The link between the data and its owner is provided by public-key certificates and attribute certificates, respectively. Certificates are particularly good to use in interconnected open systems. They can be used in applications, where a large number of entities exists and even entities associated to different institutions unknown to each other have to authenticate and communicate securely [59].

Certificates are generated and issued by a trustworthy authority named Certificate Authority (CA). The security policy of a CA describes the lifecycle of key pairs or attributes and the validity period of a certificate. A certificate to be valid must be signed with CA's private key. Therefore, certificates are unforgeable and usually securely submitted to an authority that provides certificates. In some circumstances, a certificate must to be revoked, i.e. its validity must be terminated sooner than assigned [59].

A X.509 is a format for digital certificates widely used. It contains information about the identity to which a certificate is issued and the identity that issued it. Standard information in an X.509 certificate includes [53]:

- *Version* - version of the X.509 certificate (different versions have different data that is included)
- *Serial number* - it is assigned by the identity that creates the certificate, it is used to distinguish it from other certificates
- *Algorithm information* - indicates which algorithm is used by the issuer to sign the certificate
- *Issuer distinguished name* - certificate issuer entity name (usually a certificate authority)
- *Validity period of the certificate* - start/end date and time
- *Subject distinguished name* - name of the identity that the certificate is issued to
- *Subject public key information* - the public key associated with the identity
- *Extensions* (optional)

2.7.2 Document certification

One of the most widely digital documents used are Portable Document Formats (PDFs). They support digital signatures in order to provide authentication of digital data based on Public Key Infrastructure (PKI) technologies. Organizations that want to sign their documents can use a PKI that includes an independent authority that issues, records, and tracks digital IDs.

PDFs support embedding signatures and issuer's certificate inside the document, allowing the viewing application manage the signature with a single file for each signed document. The signature is created using a hash code of the document (SHA-256 for example) and encrypted with signer's private key. Then, the signature is placed in a reserved space left for that purpose inside the document [50].

To validate a signature, the validator simply retrieves the signer's certificate and compares it to their own list of trusted certificates [50]:

1. The viewing application generates a hash of the document
2. The encrypted hash inside the document is decrypted using the signer's public key
3. The two hashes are compared
4. If they are equal, the signature is reported as authentic

2.7.3 HTTPS

Hypertext Transfer Protocol (HTTP) does not provide any kind of security, i.e. the data transmission is made in plain text. This allows intruders to read the communications and steal the information or even modify it.

To secure web-based applications we can rely on the HTTP Secure (HTTPS) protocol to provide privacy and security in transactions. To protect the communications, they are encrypted using the Transport Layer Security (TLS), formerly known as Secure Sockets Layer (SSL). Its application ranges from home banking, e-commerce and e-procurement to deal with sensitive data. Users can trust in this protocol to prevent their personal and confidential information from unauthorized view when is being transfered of over the Web [10].

But, TLS can be targeted by attacks like man-in-the-middle (MITM). It exploits the fact that the TLS server sends a certificate with its public key to the Web browser. If this certificate is counterfeit, the entire communication under risk. Such an attack replaces the original certificate authenticating the TLS server with a modified certificate. The attack is successful if the user neglects to double-check the certificate when the browser sends a warning notification. This occurs all too often, especially among users who frequently encounter self-signed certificates when accessing intranet sites [10]. Users should never trust in unknown certificates.

2.8 HTML5 Canvas

HTML5 Canvas is an HyperText Markup Language (HTML) element introduced in the recent version 5 of HTML which allows to draw inside of a web-page.

HTML5 Canvas is an immediate mode bitmapped area of the screen that can be manipulated with JavaScript. Immediate mode refers to the way the canvas renders pixels on the screen. HTML5 Canvas completely redraws the bitmapped screen on every frame using Canvas API calls from JavaScript. The programmer have to set up the screen display before each frame to be rendered in order to the correct pixels being shown.

Basic HTML5 Canvas API includes a 2D context that allows a programmer to draw various shapes, render text, and display images directly onto a defined area of the browser window. It is possible to apply colors, rotations, alpha transparencies, pixel manipulations and various types of lines, curves, boxes, text and images [20].

It is also sensible to touch events. It is possible define event listeners that will be fired to indicate that touch-related changes have occurred [19]:

- *touchstart* - event called when the user places a touch point on the touch surface
- *touchend* - event called when the user removes a touch point (lift a finger or stylus) from the surface, it is is also called when a touch point moves off the edge of the screen
- *touchmove* - event called when the user moves a touch point along the surface. The rate at which events are sent is browser-specific and the capability of the user's device
- *touchcancel* - event called when a touch point has been disrupted in some way

Each touch event can provide a list of touches, depending how many fingers are touching the screen. A touch represents just a single contact point on the touchscreen. A touch is represented by a data structure which contains several read only properties with information about it [3]:

- *identifier* - it is a unique identifier for the touch. The same touch point will have the same identifier for all movements around the surface
- *screenX* - the X coordinate of the touch point from the left edge of the screen
- *screenY* - the Y coordinate of the touch point from the top edge of the screen
- *clientX* - the X coordinate of the touch point from the left edge of the browser viewport

- *clientY* - the Y coordinate of the touch point from the top edge of the browser viewport
- *pageX* - the X coordinate of the touch point from the left edge of the document, it can include the horizontal scroll
- *pageY* - the Y coordinate of the touch point from the top of the document, it can include the vertical scroll
- *target* - the element where the touch point occurs

It is also possible to read features from the touch for pressure or force, radius in x and y and even its rotation. But, unfortunately, these are still experimental features and are not standardized. Its support is still very limited and its usage is discouraged [3].

The HTML5 Canvas is widely fully supported by the most used browsers, since the following versions [55]:

- (2011) Internet Explorer 9
- (2010) Mozila Firefox 3.6
- (2009) Safari 4.0
- (2010) Google Chrome 4.0
- (2009) Opera 10.1
- (2010) iOS Safari 3.2
- (2011) Android Browser 3.0

2.9 QR Codes

A QRCode consists in a matrix barcode, it can also be seen as a two-dimensional barcode. This allows to store more information in comparison with a one-dimensional barcode.

QRCodes have been used widely. They are easy to use and allow to attach information to something, like a product or an ad. To read the QRCode, the user should point the camera's device to the code, the device reads and processes the image and then recognizes the information [52].

A common use of QRCodes is to encode an Uniform Resource Locator (URL), which can be very useful. This enables users to easily access any website.

QRCodes are also provided with error correction, making it still readable if some parts of the code is missing. This is achieved by using Reed-Solomon algorithm.

QRCodes are an effective and very convenient way to share information between different types of media. Actually they are now present around us like in many advertising posters. Another way that is not so well exploited is to use QRCodes to share secrets between a computer and a smartphone, allowing for example to synchronize a service between a computer and a smartphone.

2.10 Web services

A Web Service is a service provided by a device and other devices can access it. They should understand the same protocol used by the Web Service and they can communicate through the Internet or over a network. There are two types of Web Services that are widely used: SOAP and REST.

2.10.1 SOAP

Simple Object Access Protocol (SOAP) is designed to provide interoperability between heterogeneous devices. It follows the architecture defined by Service-Oriented Architecture (SOA) and can be used over different protocols as HTTP and Simple Mail Transfer Protocol (SMTP). SOAP is a message format based on Extensible Markup Language (XML) used to invoke remote functions in other applications.

The applications can be running in devices with different hardware platforms, operating systems and programming languages. Since it can be used with HTTP will usually pass through firewalls.

SOAP is typically slower than other standards because of the XML parsing and latency introduced by firewalls when analyzing the SOAP packages.

A SOAP message is called envelope, which contains a header and a body. The header can contain information about routing and Quality of Service (QoS) configurations. The body contains the payload of the message. XML Schema is used to describe the structure of the SOAP message, so that SOAP engines at the two endpoints can marshall and unmarshall the message content and route it to the appropriate implementation [44].

2.10.2 REST

REpresentational State Transfer (REST) is a software architecture style to build scalable web services. The systems that follow principles of REST are called RESTful. It is the basis concept behind the HTTP protocol and it is designed based in four principles [44]:

- **Resource identification through Uniform Resource Identifier (URI)** - a RESTful web service makes his resources available through an URI to the clients. Once the resources are identified by URIs, a global addressing space for resource and service discovery is provided
- **Uniform interface** - resources are manipulated using four operations *create*, *read*, *update* and *delete*. *Put* creates a new resource, *delete* removes a resource, *get* retrieves the state of a resource and *post* transfers a new state of a resource
- **Self-descriptive messages** - resources are independent from their content format (HTML, XML, pain text, images, etc.). Meta-data about the resource can be used to control caching, detect transmission errors, negotiate the appropriate representation format and perform authentication or access control
- **Stateful interactions through hyperlinks** - every interaction with a resource is stateless, all information necessary to fully understand a request is sent with the request. Several techniques exist to exchange state, like URI rewriting, cookies, and hidden form fields

RESTful Web Services are commonly adopted. It requires a small infrastructure and most of HTTP servers already provide it. The clients are easy to build because it can be used with an ordinary Web Browser installed on the client. It is possible to scale RESTful Web Services to serve a large number of clients by resorting to caching, clustering and load balancing.

Meanwhile some misconceptions are made regarding to the REST principles. Some of the operations are blocked by firewalls and only *get* and *post* methods are used in HTML forms, leading to a series of workarounds implement similar functionalities that are provided by REST.

Chapter 3

Implementation

In this chapter we will describe how we have implemented the biometric system for handwritten signature verification and some of the problems found during work development. The system will be capable of collecting the signatures from the users and compare those signatures to verify the authenticity. As mentioned in the introduction, the system will have three components: a web service which will provide the service, a client which will be the interface between the web service and the user and a mobile device which will acquire the signatures.

The biometric handwritten signature is acquired from the mobile device and converted into a set of time/space points. That can then be compared with previously enrolled biometric markers of the user's handwritten signature. The goal of this work is to provide user authentication that is sufficient for majority of online user authentication and digital documents authenticity with the collection of these simple biometric features.

The first time the user uses the system, he have to enroll his signature many times in order to train the system. From those signatures is created the reference signature. When is necessary to verify the authenticity of a user, a new signature is gathered and compared with the reference signature.

Firstly, we will present the general architecture of the system and then we will define what a signature is and how they are acquired from the the mobile devices. After, we will talk about the implemented operations which are necessary in the system, namely to compare signatures. Then we will address how we have implemented the enrollment system and from there how we build the reference signature. Only then we have the necessary tools to perform the comparison of the signatures. Lastly, we talk about some technical details important in the system about the communication and storage.

3.1 System architecture

On Figure 3.1 we can see a general overview of the system architecture with the interactions between each component. The system is composed by three main components:

- **Authenticity Service server** - is responsible to coordinate all actions between the components, provide a web service, store all information and offer authentication
- **Client** - is the interface between the authenticity service and the user and is also used to interact with the mobile device trough QRCodes
- **Mobile device** - is used to collect the signatures

The following steps show the general procedure to collect a new handwritten signature every time is necessary:

1. The user starts the process by requesting a new ID for a signature
2. The signature ID is generated and returned from the sever to the client
3. The client shows a QRCode which encodes an URL. The URL is a link to the mobile application and contains the signature ID
4. The user, with camera of the mobile device, reads the QRCode and opens it. Then, the application is shown and the user can sign with his handwritten signature. The application is basically a HTML5 application which contains a Canvas
5. After the signature to be acquired, it is sent to the Authenticity Service server
6. Finally, the result is sent back to the client for further actions

The signature IDs are randomly generated identifiers. They identify uniquely a signature and each signature is represented by one. In the last step of the process, the result sent from the server is different depending on the intended purpose. For

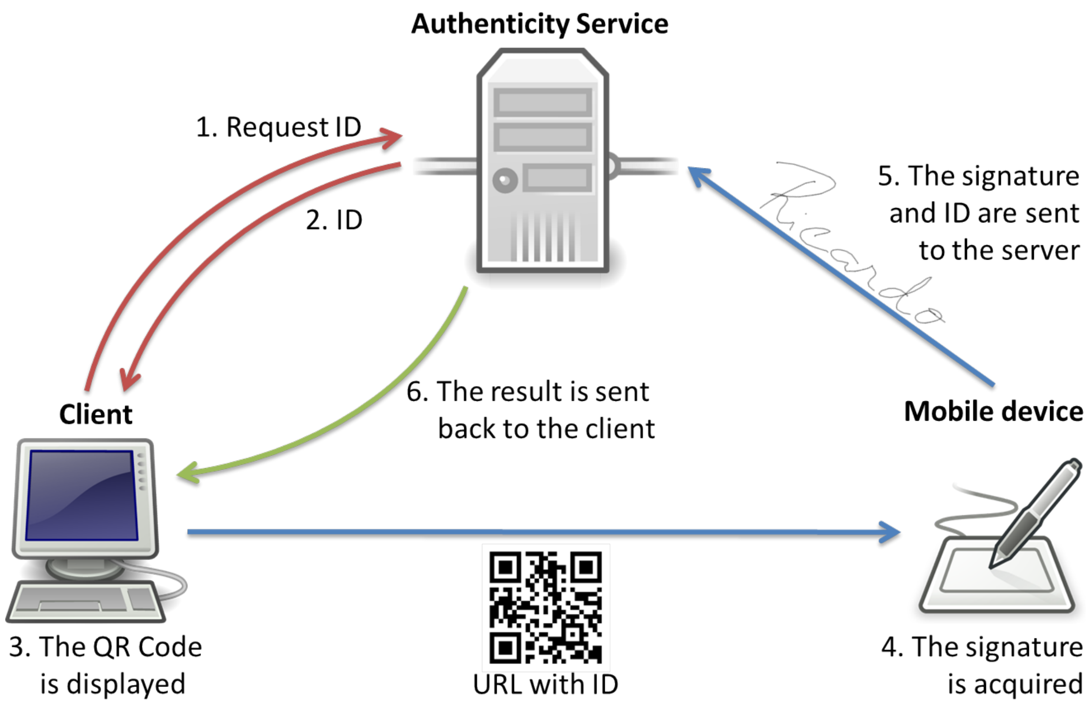

Authenticity Service
1. Request ID
2. ID
5. The signature and ID are sent to the server
Ricardo
Client
6. The result is sent back to the client
Mobile device
3. The QR Code is displayed
URL with ID
4. The signature is acquired

Figure 3.1: Architecture overview of the system

example, it can be a notification if it is a valid signature, a digitally signed document with the handwritten signature or even redirect the user to a new page after a successful login.

3.2 Signature definition

The signatures are defined as follows:

- **Signature** - is a set of disconnected segments
- **Segment** - is a set of connected points, a segment starts when the user starts writing and ends when he lifts up the stylus from the device
- **Point** - is obtained every time the mobile device makes readings from the touchscreen, each point has coordinates for *time*, x and y

The points are represented in a Cartesian coordinate system with two dimensions (x, y). Only the positive part is considered and for convenience it is inverted around

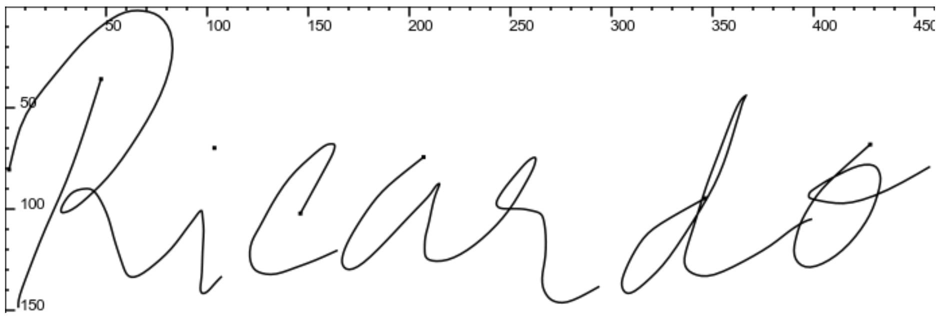


Figure 3.2: Representation of a coordinate system with a signature

Signature	
Segment	Data
1	$\{(0,104,136),(120,111,152),(137,116,161),(154,120,169)\}$
2	$\{(663,227,128),(736,224,145),(754,221,160),(770,220,171)\}$
3	$\{(1132,278,142),(1203,280,163),(1220,282,170),(1238,284,177)\}$
4	$\{(1425,252,142),(1537,268,134),(1554,274,132),(1571,280,131)\}$

Table 3.1: Example of the data structure of a signature

the xx axis. The points also have the sampling time associated. Figure 3.2 shows an example of a signature in respective coordinate system represented.

Table 3.1 is an example of the data structure of a signature. There is a set of segments (4 in this case) and each segment is a set of points (each segment has 4 points in this case).

3.3 Signature acquisition

The touchscreens of smartphones and tablets are capable of collecting handwritten signature images and associated biometric markers derived from the motion direction of handwritten signatures. These time based biometric markers are then converted into signaling time waves, by using the dragging or lifting movements the user makes with a touchscreen omnidirectional tip stylus and then converted into a biometric bit stream. This biometric data can be matched with previously enrolled biometric markers.

But mobile devices are not specialized devices to collect handwritten signatures, however they are widely spread and much them in good condition for signature

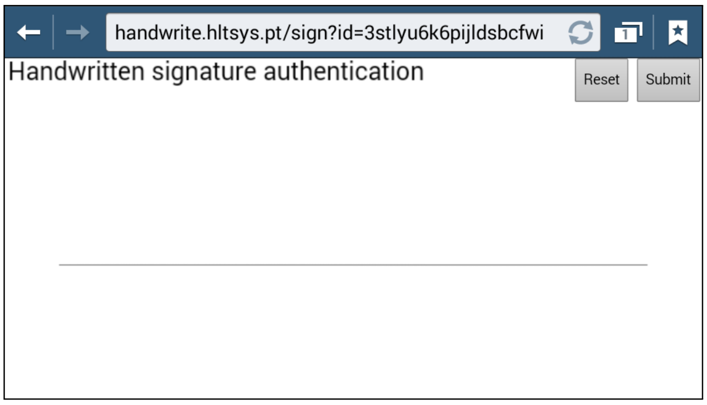

handwrite.hltsys.pt/sign?id=3stlyu6k6pijldsbcfwi
Handwritten signature authentication
Reset
Submit

Figure 3.3: Screenshot of the smartphone application

collection. We therefore focused on developing a HTML5 application capable of running on most of the devices of this type. So, we created a small web application built in HTML5 and JavaScript. The application is mainly a HTML5 canvas which resorts to three events: *touchstart*, *touchmove* and *touchend*.

The first time a *touchstart* event is detected, the application stores the current system time. Every time this event occurs, it will mark the start of a new segment in the signature. The *touchend* event will mark the end of the segment. Between these two events (*touchstart* and *touchend*), the *touchmove* event will occur everytime that a movement is detected.

The values stored for position are the coordinates where the touch occurs, i.e the number of pixels from the left and from the top of the screen. The sampling time for each point is obtained according to the following formula, where P_t is the sampling time of the point, T_c is the current time and T_s is the time when the signature was started:

$$P_t = T_c - T_s$$

For simplicity, all times are stored in milliseconds, allowing the easy application of algebraic operations.

Figure 3.3 shows the appearance of the application in a mobile device running the Android system.

3.4 Signature operations

There are many the operations to make the necessary manipulations which enables ultimately the comparison of signatures. The operations for normalization and the time adjustments create a better overlap between signatures. The interpolation of segments allows to estimate new points which were unknown. Lastly we will talk about how the speed and acceleration are calculated to help in the comparison and how the signatures are smoothed to have a better look.

3.4.1 Normalization

We need to “normalize” signatures before starting to do operations on them. The aim of normalization is to create a correct overlap between the signatures both for creating the reference signature and for compare signatures, ensuring the operations consistently applied to the signatures. To normalize the signatures, two procedures are applied: translation to origin and scaling.

3.4.1.1 Translation to origin

In order for the operation be applied properly, signatures should be translated to the origin. By doing this, the user can sign anywhere on the screen of the smartphone. Otherwise, the place where the user signs could have implications on the result.

Saying translation to origin means that the leftmost point of the signature should have the value 0 on x-axis and the topmost point of the signature should have 0 on y-axis.

To achieve this, the following procedure can be used:

1. Find the leftmost point
2. Find the topmost point
3. Subtract each point with the leftmost point only on x-coordinates
4. Subtract each point with the topmost point only on y-coordinates

After executing this procedure, the entire signature is moved. The information about the location where the user has signed was lost.

3.4.1.2 Scale

Similarly to translation to origin, it is also important to scale the signatures. This allow to the size which the user signs do not influence the process. If the user signs with a device with a larger screen than the device he usually signs, he will naturally make the signature larger also. Even so, if he uses always the same device, he can sign with different sizes.

This process involves two signatures, one will be the signature that we want to normalize and the other will be used as reference. Then we have to find two scaling factors, one for x-axis and the other for y-axis, i.e. it is a distorted scaling because we have different scalings. Those factors are calculated with the rightmost and the bottom-most points of both signatures. Signatures should be moved to the origin before we find these points. Otherwise, if we do not move the signatures to the origin, they will not end up with the same size. The order in which operations are executed is also important.

To scale the signature, the following steps are performed:

1. Translate both signatures to the origin
2. Find the rightmost and the bottom-most points of both signatures
3. Calculate the scaling factor for x and y-coordinates by using the following formula, where R_R and R_B are the rightmost and bottom-most points of the reference signature, S_R and S_B are the rightmost and the bottom-most points of the other signature and C is the scaling factor:

$$C_x = \frac{S_{R_x}}{R_{R_x}}, C_y = \frac{S_{R_y}}{R_{R_y}}$$

4. Multiply each point of the signature by C_x on x-axis and for C_y on y-axis.

After executing this procedure, the entire signature is scaled to the pretended size. The information about the size and position where the user signed is lost. Actually, the translation could be done during the signature acquisition but we loose that information permanently and that disables the study about if the location have any biometric meaning.

3.4.2 Time adjustments

Before we make the comparison of two signatures, we do some time adjustments to improve the comparison. When the user signs many times, he will not sign exactly in the same way, he will make some variations in space and time.

Yet, despite the same user user signs in a similar way, the differences in time can affect considerably the comparison. Users tend to have small differences in time leading to a critical misalignment. The difference is around 200ms in the end of the signature according to experimental results, with the dataset that we have collected. For example, in the same sampling time, in one signature the movement can be upward while in the other signature can be downward. If we make comparisons based solely on distance we will get wrong results.

The approach implemented to solve this problem was to synchronize the segments in time, i. e. modify the segments to start and end at the same time. The points of the segment are scaled in time according to the next formula, where λ is the scaling factor for a determined segment, R_s and R_e are the starting and the ending times of a segment from the reference signature and S_s and S_e are the starting and the ending times of the signature that we want to adjust:

$$\lambda = \frac{R_e - R_s}{S_e - S_s}$$

And then change the points accordingly with the factor found for that segment, where P_r is the resulting time and P_t is the original time of the point:

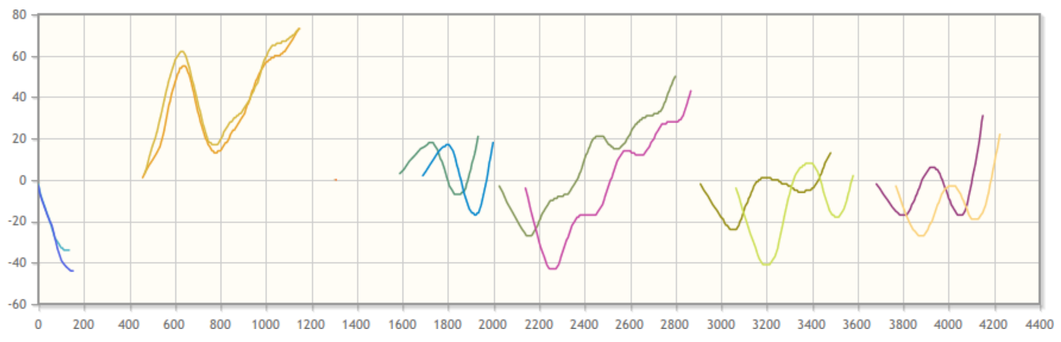
$$P_r = (P_t - S_s)\lambda + R_s$$

Figure 3.4 shows an example of comparison between two signatures, with and without time adjustments. Without aliments we can see the misalignments happening over time where we can find points where the movements are opposite.

When comparing two signature we must have in consideration these time differences, we cannot discard that information. This is useful in the comparison because we can be in a situation of a forgery.

3.4.3 Interpolation of segments

Sometimes it is useful to estimate new points between two adjacent points. We can use interpolation to find the new points. Interpolation is a mathematical method to


(a) Without time adjustments

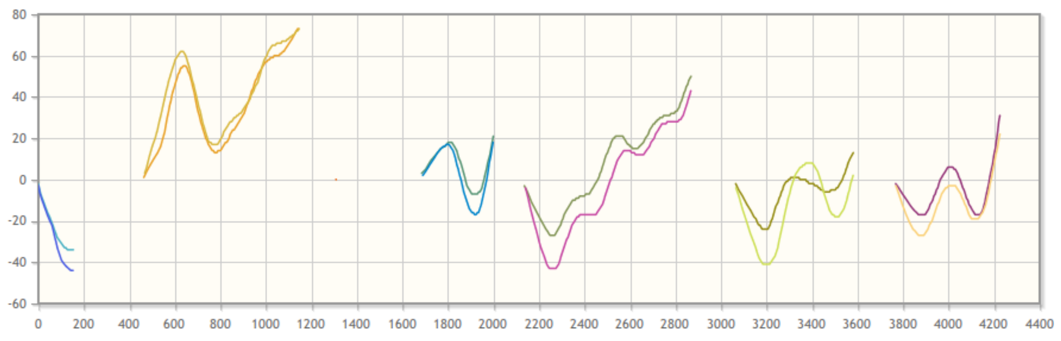

(b) With time adjustments

Figure 3.4: Example of misalignment of segments without time adjustments

construct new data points from discrete known sets. This is necessary to compare despite the signatures do not have the same number of points.

Next, we will show a linear interpolation method to estimate a new point. Figure 3.5 illustrates that scheme with the point represented. Each segment will be parametrized in the interval $[0, 1]$ and α will the position in that interval of the new point.

Let's start by defining δ. It is the difference in percentage between two adjacent points and can be written as the following formula, where n is the number of points of the

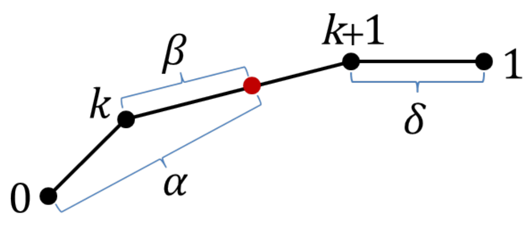


Figure 3.5: Scheme of the linear interpolation method, the new point is marked in red

segment:

$$\delta = \frac{1}{n-1}$$

k and $k+1$ are the indexes of the points before and after from where α is located in the segment:

$$k = \lfloor (n-1) \cdot \alpha \rfloor$$

β is the percentage between the two points of the place where α is:

$$\beta = \frac{\alpha - \delta \cdot k}{\delta}$$

We can now find the interpolated value. This calculation is valid for time and for x and y coordinates. R is the interpolated point, P_k and P_{k+1} are the points before and after the estimated point:

$$R_t = (P_{k+1_t} - P_{k_t})\beta + P_{k_t}, R_x = (P_{k+1_x} - P_{k_x})\beta + P_{k_x}, R_y = (P_{k+1_y} - P_{k_y})\beta + P_{k_y}$$

It is also possible to use an approach using a polynomial interpolation of higher degree or a spline interpolation. We have implemented a spline method to smooth the signatures and we could benefit from it, but this method was used before that implementation and we kept using it because we did not have time to modify the system. In practice, it will not change very much the result because the variations are small and with low relevance. On the other hand, the linear interpolation is computationally easier.

3.4.4 Speed and acceleration

The first approach to compare the speed and the acceleration of the signature was to calculate the derivative of two consecutive points. Resorting to the definition of derivative:

$$\lim_{h \to 0} \frac{f(a+h) - f(a)}{h}$$

we can apply this formula to the discrete domain. Then we obtain the following formula, where S is the speed and P_0 and P_1 are the two consecutive points in a segment:

$$S_x = \frac{P_{1_x} - P_{0_x}}{P_{1_t} - P_{0_t}}, S_y = \frac{P_{1_y} - P_{0_y}}{P_{1_t} - P_{0_t}}$$

To obtain the acceleration, we can apply the same calculations but instead of being applied to the raw points, apply to speed. A is the resulting acceleration:

$$A_x = \frac{S_{1_x} - S_{0_x}}{S_{1_t} - S_{0_t}}, A_y = \frac{S_{1_y} - S_{0_y}}{S_{1_t} - S_{0_t}}$$

The first time when this method was ran, the results were similar to noise as shown in figure 3.6(a). This kind of results does not provide any useful information for the comparison.

To try to solve this problem we implemented signature smoothing procedure as described in the next section *3.4.5 Signature smoothing.* The reason we opted for such solution has to do with the fact that we can derive a function instead of calculating the difference between points. Adopting this solution we expected smooth values for speed and acceleration, but the improvements were none.

Later we discovered the reason why this was happening and it was due to the high sample rate added with small bugs in the calculations. When calculating the average signature, the number of resulting points were the sum of the number of points of the signatures used to build the reference signature. As the number of points is very high the distance between them is very small, which makes the slope very susceptible to variations.

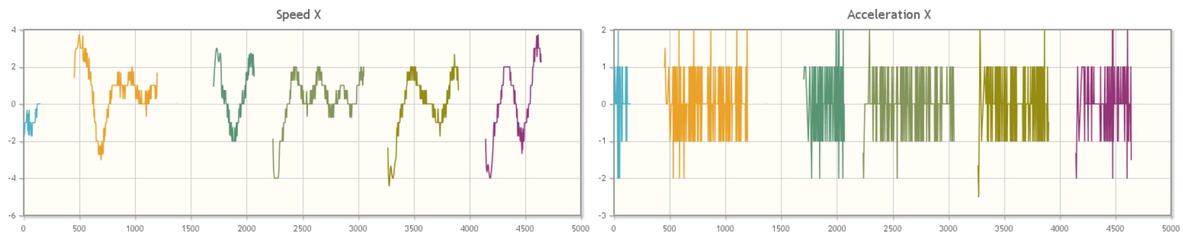
The solution was that we found was reduce the number of points. Instead the resulting number of points to be the sum of all points of the gathered signatures in the enrollment, we changed to the average number of points. This increased the space between points enabling to extract useful information. Figure 3.6 shows the effect of this change. As we can see, we got nice values for speed and acceleration.

3.4.5 Signature smoothing

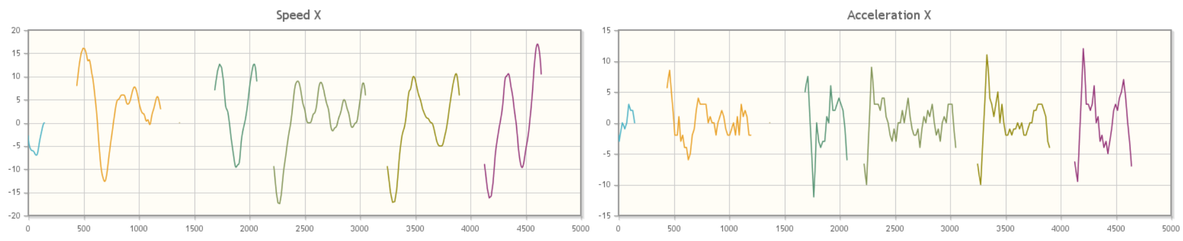
In the previous section we explained the reason to implement signature smoothing in the system. The goal was to obtain good results for speed and acceleration, but later it was revealed that was not necessary. Once implemented, it becomes an advantage of having smooth signatures. One of the advantages is the visual enhancement.

We can see an example in figure 3.7 where is shown a signature without and another with it. The difference is slight but with a closer look is noticeable.

Smoothing is achieved by constructing a interpolating high degree polynomial function from a set of points. That function is called a B-spline and allows derivatives with

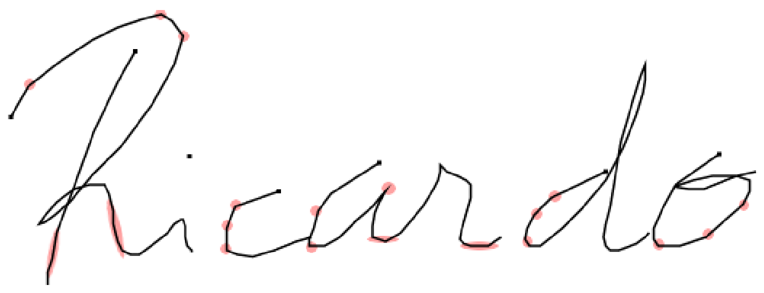


(a) Wrong sample rate

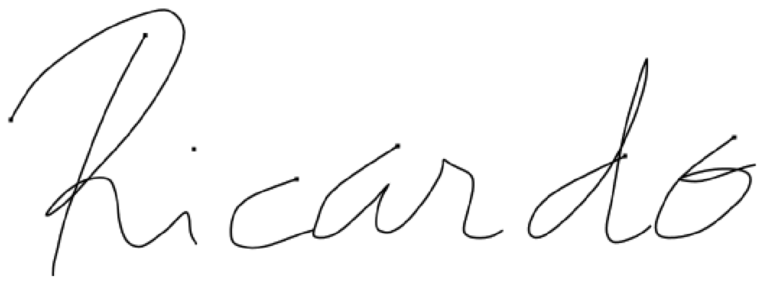


(b) Correct sample rate

Figure 3.6: Effect of high sample rate when calculating the derivative of discrete values



(a) Signature without smoothing



(b) Signature with smoothing

Figure 3.7: Effect of smoothing the signature, the regions where the smoothing is more prominent are highlight in red

C^2 continuity, i.e. the derivatives until the second degree are continuous. It is interesting to use this tool because we can obtain the speed and the acceleration from the derivative.

More than the visual effect, the calculations for comparison are also based in these results. Even the speed and the acceleration are smoothed. Instead of using the method described in the previous section, we benefit of having a derivable function to automatically smooth speed and acceleration.

B-splines are related with Bézier curves [1]. First we will introduce the Bézier curves and then how B-splines are built. We also discuss about their derivatives which are important for calculations.

3.4.5.1 Bézier curves

Bézier curves are parametric curves. The most usual cases are linear, quadratic or cubic Bézier curves, but we can define curves with higher dimensions. We will use cubic curves because they provide C^2 continuity.

These curves are intended to connect two points according with the control points. The cubic Bézier curves are modeled with four control points. The curve starts in the first point and ends in the last point. The other two points will control the shape of the curve.

The Bézier curves are parametrized in [0,1] and are defined as follows:

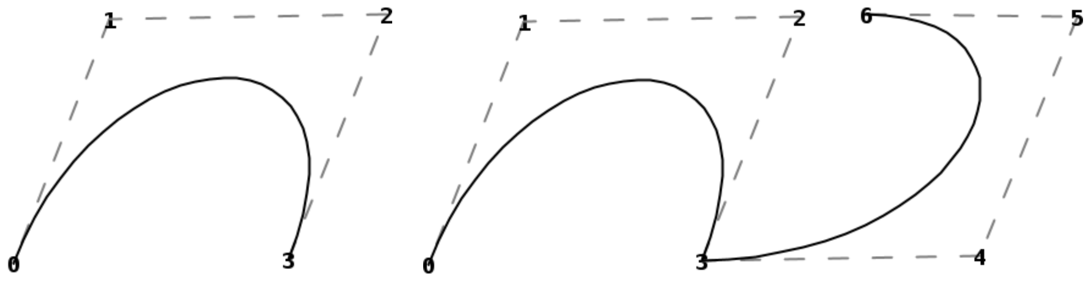
$$\mathbf{B}(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t)t^2 P_2 + t^3 P_3, 0 \leq t \leq 1$$

where P_0, P_1, P_2 and P_3 are the control points.

Figure 3.8(a) shows an example of a Bézier curve with 4 control points and figure 3.8(b) shows what happens if we attach two curves with a shared point. To solve the problem of connecting two curves we have to resort to B-splines curves.

3.4.5.2 B-spline curves

B-splines stands for basis spline. A spline is a piecewise function defined by polynomials. These functions are characteristic to have a high degree of smoothness where the several pieces connect.



(a) A single Bézier curve (b) Two Bézier curves simply connected

Figure 3.8: Examples of Bézier curves

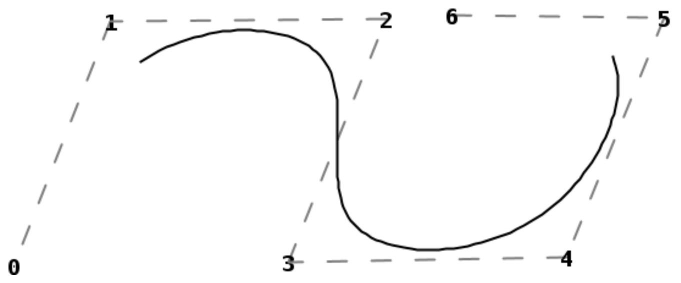


Figure 3.9: An example of a B-spline

The basis function for a cubic B-spline is [30]:

$$\mathbf{S}(t) = -\frac{1}{6}(t-1)^3 P_0 + \frac{1}{6}(3t^3 - 6t^2 + 4)P_1 - \frac{1}{6}(3t^3 - 2t^2 - 3t - 1)P_2 + \frac{1}{6}t^3 P_3, 0 \le t \le 1$$

The 1st derivative of basis function for a cubic B-spline:

$$\mathbf{S}'(t) = -\frac{1}{2}(t-1)^2 P_0 + \frac{1}{2}(3t-4)tP_1 - \frac{1}{2}(t-1)(3t+1)P_2 + \frac{1}{2}t^2 P_3, 0 \le t \le 1$$

The 2nd derivative of basis function for a cubic B-spline:

$$\mathbf{S}''(t) = (1-t)P_0 + (3t-2)P_1 + (1-3t)P_2 + tP_4, 0 \le t \le 1$$

where P_0, P_1, P_2 and P_3 are the control points. But we still have only 4 control points. We can iterate over all points of the signature and apply the formulas to the last 4 points. By doing that we obtain a curve as exemplified in figure 3.9.

The time is not derived, we should use the basis function of a cubic B-spline instead.

3.5 Enrollment process

Before the user can start using the system, he must enroll in the system. To guide the user in this process we have developed a wizard.

To start the user has to provide a username which is unique and identifies the user. It also has a captcha to protect the system from being "spammed" with fake data. The user is thus asked to produce five similar signatures. This value was determined with experimental results, not only to train the system but also to train the users to reproduce the same signature.

Finally, the user is requested to set a password that he can use to recover his account if some problem occur when using the signature to login. Then the registration is complete and the user can login into his account.

3.5.1 Screening of signatures

After the user signs the five signatures, the system will make a simple test ensuring that all signatures are in some way similar.

It happens sometimes, users do not always do the signatures in the same way. With this process the user is compelled to sign in the same way, helping him to remember how he had signed. It also allows to discard signatures that are incorrect, e.g. the user made an intentional touch.

The algorithm used to find which are the signatures that are not similar follows this procedure:

1. Compare all signatures between them, this means that for each signature compare it with all others signatures
2. Find the set with more similar signatures (we will describe bellow)
3. Discard signatures that does not belong to the set found in the previous step

When this procedure is complete, the system will ask the user to repeat the signatures that were discarded. When done, the process is repeated until the five signatures are similar.

Signats.	**1**	**2**	**3**	**4**	**5**
1	▨	✓	✓	✓	×
2	▨	▨	✓	✓	✓
3	▨	▨	▨	✓	×
4	▨	▨	▨	▨	✓
5	▨	▨	▨	▨	▨

Signats.	**1**	**2**	**3**	**4**	**5**	**Count**	**Keep**
1	✓	✓	✓	✓	×	4	×
2	✓	✓	✓	✓	✓	5	✓
3	✓	✓	✓	✓	×	4	×
4	✓	✓	✓	✓	✓	5	✓
5	×	✓	×	✓	✓	3	×

Table 3.2: Example of a result of similarity between five signatures, illustrating how the screening algorithm works

The comparison between the signatures produces a table like the example on table 3.2. It is a $n \times n$ table (in this case is five) indicating which signatures are similar. For example, if we pick the signature 2 from the left column and the signature 4 from the top, we learn they are similar. But if signature 2 is similar with signature 4, the opposite is also true. So we can take the upper triangle of the matrix and reflect it to the lower triangle. And of course, a signature is similar with itself.

After the upper triangle is computed, the remaining table is easily derived. Then, to find largest set of similar signatures, we count the number of matches for each row in the table. The rows which does not have the same number of matches of the maximum rows are the signatures that are not in the largest set and are discarded. When the maximum is 1 means that all signatures are different.

Back to the example of the table 3.2, we can see the countings per row and the rows with the maximum are highlighted. Those are the signatures that are kept and the others are discarded because they do not belong to the set with more similar signatures.

This procedure is not very efficient because requires $O(n^2)$ comparisons, it does not scale very well. But since we only have five signatures it necessary to make 25 comparisons in maximum. In fact, only 10 comparisons are made because some values are derived as described previously.

3.6 Reference signature construction

During the enrollment process, the system asks the user to sign five times. Those signatures are used to create a signature that will be used later in the comparison, which will be called as reference signature. The reference signature is built using the average of the signatures gathered during the enrollment.

To calculate the average signature we start by getting the number of average points by segment, as referred in the section *3.4.4 Speed and acceleration*. Each point is obtained from the average point of the signatures acquired in the enrollment. The points used to calculate the average point are obtained using the interpolation method in its corresponding position. Figure 3.10 exemplifies that process.

Before computing the average signature, is necessary to make some adjustments to the signatures gathered in the enrollment:

- Verify if the signatures have the same number of segments
- Normalize the signatures to the maximum time, width and height
- Find the average number of points per segment

Because we are calculating the average points, it is not necessary to do the time adjustments. In this case, the average ends up canceling the effect that we address with the time adjustments.

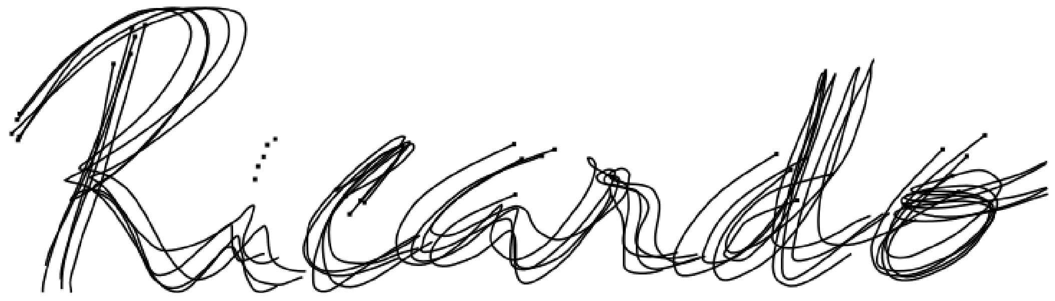
The process to calculate the average signature is described as follows. Let n be the number of signatures and N_i be average number of points in the segment i:

- For every segment i
- Create N_i points:
  - For each $k = 0..N_i - 1$,
  - Interpolate the n signatures in the segment i at position $\frac{k}{N_i - 1}$, the resulting points are $P_{1..n}$
  - The resulting average point R_k is:

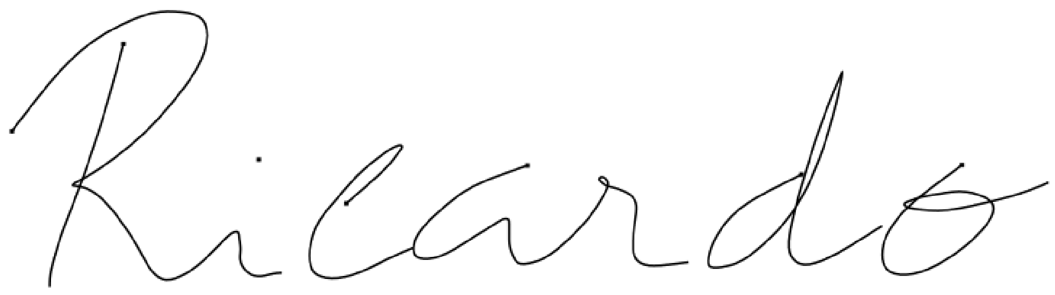
$$R_{k_t} = \frac{\sum_{j=1}^{n} P_{j_t}}{n}, R_{k_x} = \frac{\sum_{j=1}^{n} P_{j_x}}{n}, R_{k_y} = \frac{\sum_{j=1}^{n} P_{j_y}}{n}$$

  - Add R_k to the reference signature in the segment i

Figure 3.10(a) shows an example of the signatures gathered during the enrollment. The resulting average signature after this process can be seen in figure 3.10(b). Figure 3.11 shows the changes in the graphics after this procedure.



(a) The gathered signatures together



(b) The resulting average signature

Figure 3.10: Example of apply the average method to obtain the reference signature

3.7 Signature comparison

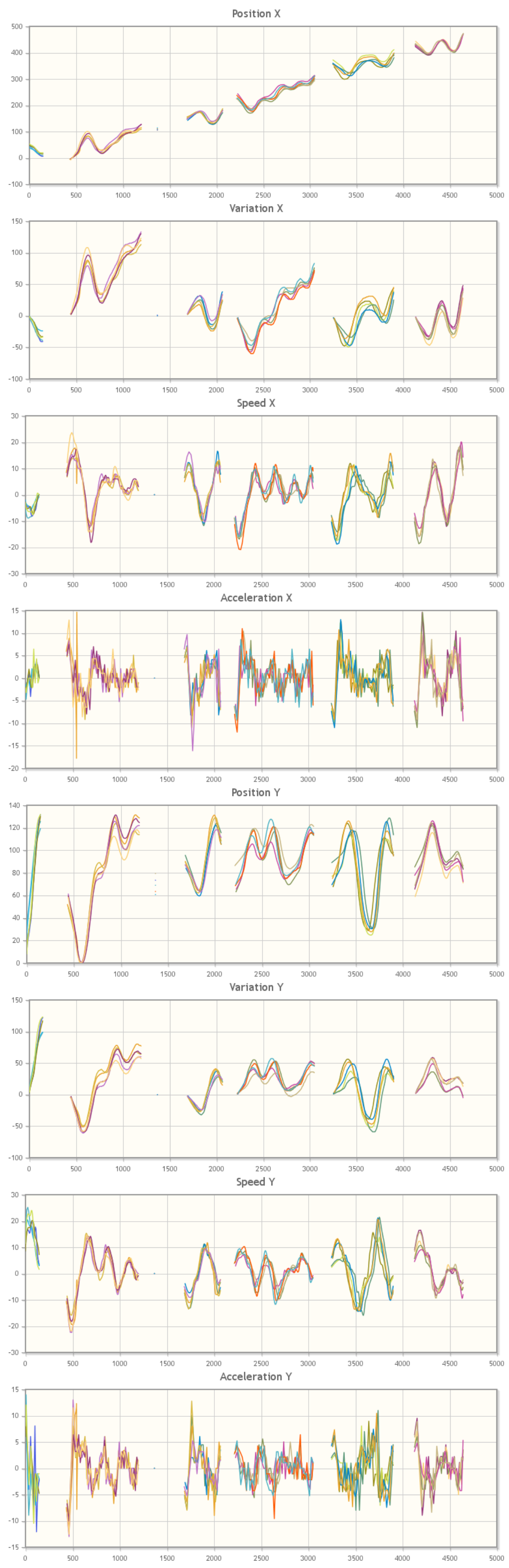
The comparison of two signatures is made by segments implying that both signatures should have the same number of segments (in future implementations, we should consider to implement a system not so rigid). Then a series of assertions are made for each point of the segments. In this case we are using conditions that will be evaluated to a boolean value and are the in the format *measurement* $<$ *threshold*. But any other conditions can be used.

From now on in this section we will refer the two signatures as S_R and S_C, the reference signature and the signature that we want compare respectively.

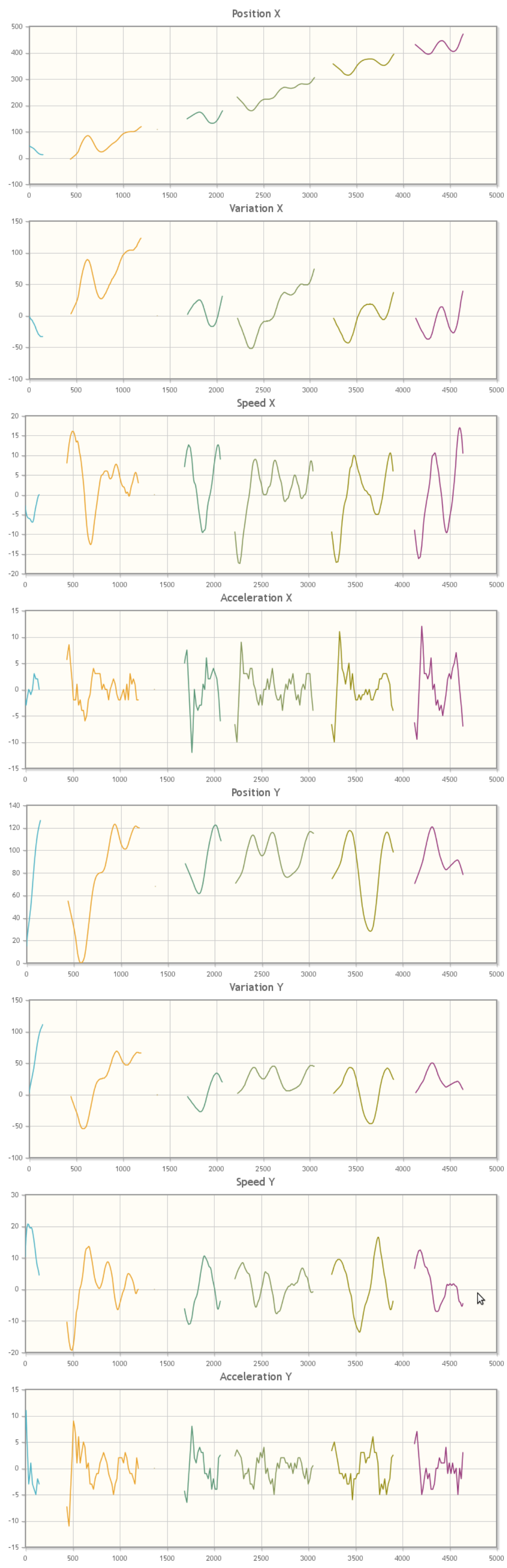
A series of parameters can be configured as thresholds to the whole signature. We will describe how they are configured in the chapter *4 Testing and Results*:

1. **Time Segment Offset** - maximum time difference that a segment of S_C can start or end in comparison with corresponding segment in S_R

2. **Time** - maximum difference of time in the B-spline curve between a point of S_R and the corresponding point in S_C



(a) Graphics of the gathered signatures



(b) Graphics of the resulting signature

Figure 3.11: Example of the graphics before and after applying the method to obtain the reference signature

3. **Distance** - maximum distance in the B-spline curve between a point of S_R with corresponding point in S_C
4. **Time1st** - maximum difference of time in the 1st derivative of B-spline curve between a point of S_R with corresponding point in S_C
5. **Distance1st** - maximum distance in the 1st derivative of B-spline curve between a point of S_R with corresponding point in S_C
6. **Time2nd** - maximum difference of time in the 2nd derivative of B-spline curve between a point of S_R with corresponding point in S_C
7. **Distance2nd** - maximum distance in the 2nd derivative of B-spline curve between a point of S_R with corresponding point in the S_C

Before the comparison can take place, we need to:

- Verify if the number of segments of S_R are equal to S_C
- Verify the difference between S_R and S_C of starting and the ending times of each segment do not exceed the threshold Time Segment Offset
- Normalize S_C by translating to the origin and by scaling to the same size of S_R
- Apply the time adjustments for S_C and use S_R as reference

Then we can finally compare the signatures. The procedure follows the steps described below:

- For every segment in S_R and
- For every point of the selected segment of S_R (it will be P_1),
- Interpolate the corresponding point to P_1 in S_C (it will be P_2) and
- Compare P_1 and P_2 with the following assertions, where V_1 and V_2 are the corresponding points of P_1 and P_2 on speed and A_1 and A_2 are the corresponding points of P_1 and P_2 on acceleration, the thresholds are also used:
  1. **Time**: $|P_{1_t} - P_{2_t}| < time$
  2. **Distance**: $\sqrt{(P_{1_x} - P_{2_x})^2 + (P_{1_y} - P_{2_y})^2} < distance$

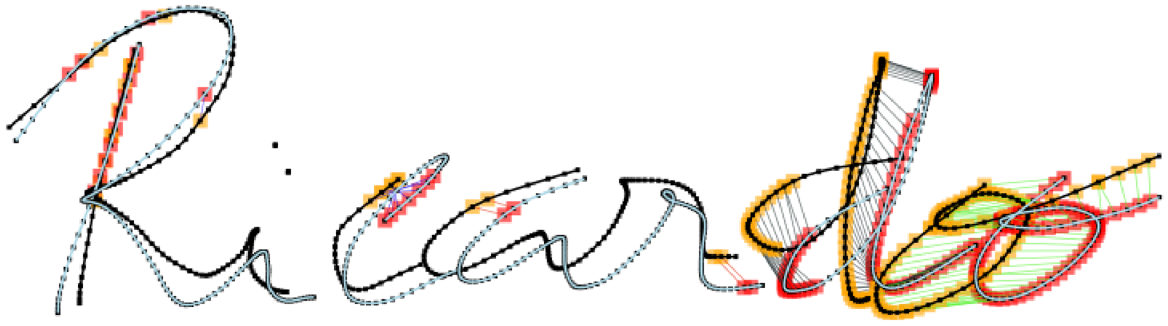


Figure 3.12: Example of a failed comparison

3. **Speed (time)**: $|V_{1_t} - V_{2_t}| < time1st$

4. **Speed (distance)**: $\sqrt{(V_{1_x} - V_{2_x})^2 + (V_{1_y} - V_{2_y})^2} < distance1st$

5. **Acceleration (time)**: $|A_{1_t} - A_{2_t}| < time2nd$

6. **Acceleration (distance)**: $\sqrt{(A_{1_x} - A_{2_x})^2 + (A_{1_y} - A_{2_y})^2} < distance2nd$

If some point fails a condition, the hole comparison will give a negative result. Figure 3.12 shows an example of two signatures that do not match. Let's say that the reference signature is S_R in lightblue and we are comparing with S_C in black. The points where some of the assertions failed are highlighted in red in S_R and in orange in S_C. It is also possible to see some guidelines connecting the corresponding points between S_R and S_C. This help to understand the locations where the comparison failed.

This method is still incomplete due to implementation time constraints, i.e. it still requires further research to get better optimization. An improvement to be made is give different importance to the conditions, for example a condition can fail in 10 points, but other can only fail in 3 points. Another improvement is regarding to the time used in the conditions 1, 3 and 5 which is the same, i.e. in the three cases the time used is from the basis function of B-spline. In fact, is also possible to derive the time but results are still unknown. To compare the distances we are using the Euclidean distance due to its simplicity and also because proved to be effective. Other types of distances can be used, for example, Manhattan distance, Canberra distance or even Mahalanobis distance [49].

Parameter	Value
id	`5nqtd6ef9ei00lr7hkvi`
points	`[ [[0,621,30],[11,621,36],[24,621,43],[35,621,54]], [[619,603,30],[630,609,28],[643,616,26],[653,625,26]] ]`

Table 3.3: Example of a message sent by the smartphone

3.8 Technical details

The server was deployed using the applicational server Apache Tomcat and implemented in Java. It uses the Java Servlets to receive the HTTP requests. The client's and mobile device's applications run in a web browser and were built using HTML technology and the dynamic parts were implemented in JavaScript.

Following we will describe how the components of the system communicate and how the storage is made in the server.

3.8.1 Messages communication

In the state of the art we introduced the concept of REST and SOAP. REST is an easy to use and widely implemented method. Because we are using browsers in the client and in the mobile device, it is transparent its usage. The web server is oriented to handle REST, what makes it also very easy to use. Some studies [41, 44, 4] suggest that REST is better than SOAP in several ways, namely in performance and ease to use. For those reasons we use REST to exchange messages between all components of the system.

The messages between the server and mobile device and the client are in JavaScript Object Notation (JSON) format. This is a format widely used and supported. To handle JSON on the server we used a library called *Java JSON*. It is easy to use and it is one of the libraries recommended by the standard (`http://json.org/`). In the browser (mobile device and client) we used the parser built-in in the browsers libraries.

On table 3.3 we can see an example of sending a signature to the server. There are two parameters: ID and points. ID is a signature identifier and points is the data representing the signature.

In the mobile device application, the data regarding the signature is stored in a

JavaScript array. The parameter points is obtained after applying the method `JSON.stringify()` in that array because is very easy to use, even to handle the data in the server. That method converts a JavaScript value to a JSON string. The array follows the next structure:

- The array represents a signature
- Each element in the signature array represents a segment
- Each segment is an array of points
- Each point is an array with three elements: `time`, `x` and `y`

So, on the example of table 3.3 the data of the parameter points represent a signature with 2 segments and each segment contains 4 points.

When the signature is displayed on the client, the server answers with a JSON object containing the signature (with the same structure as described previously), total variation, position on x, variation on x, speed on x, acceleration on x, position on y, variation on y, speed on y, acceleration on y and information about the total time took to sign, the total time actually drawing the signature, number of segments, number of points and the number of points per second. An example is shown in figure 3.13.

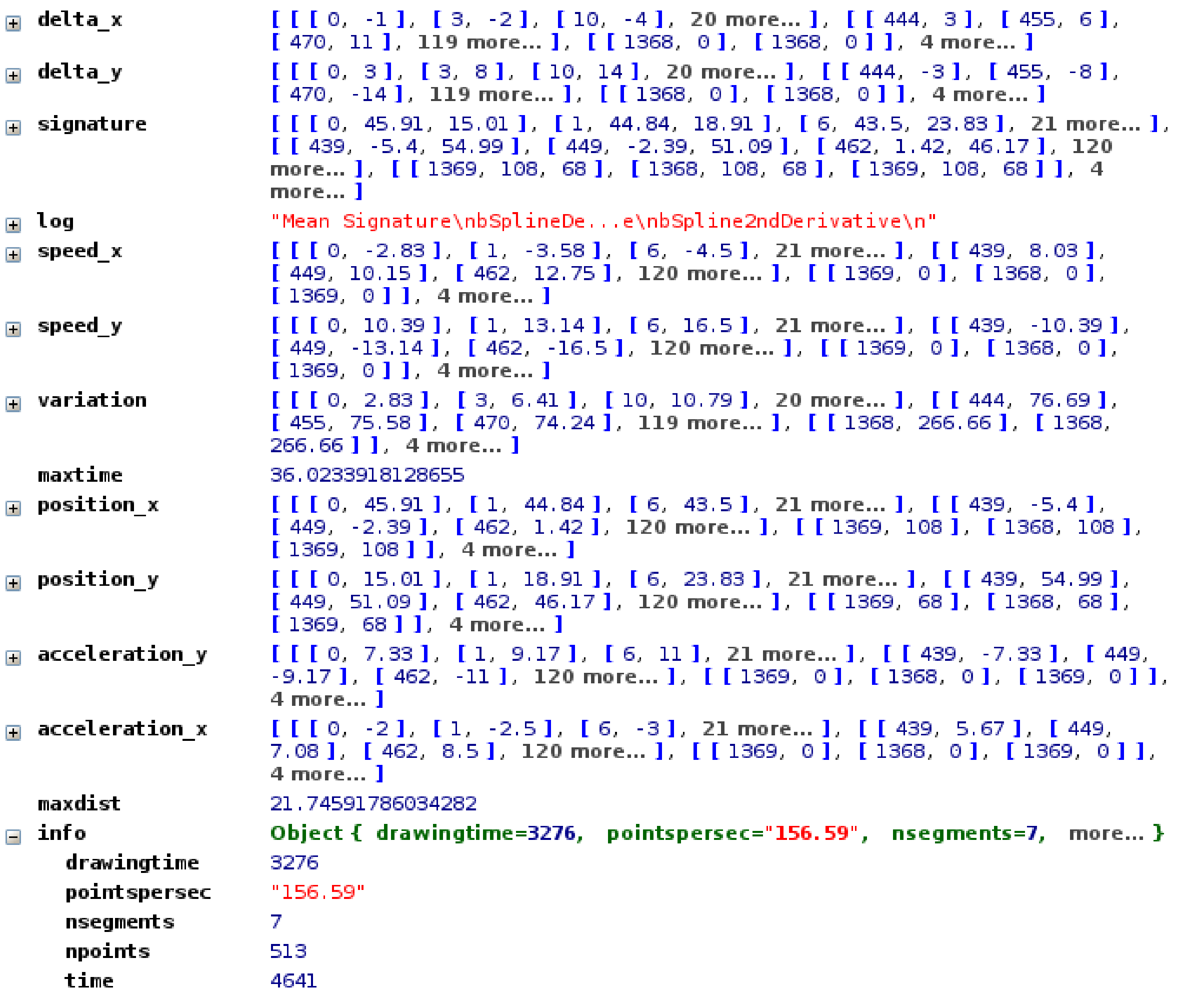
3.8.2 Database storage

All information regarding the users and its signatures has to be stored. To do that, we developed a simple but effective way to organize the information.

The database of the system is stored by using files. It is possible to configure the filesystem folder the system will use as storage. That path has no special requirements, it should be just a simple empty folder with writing permissions.

Each user is represented by a folder inside the specified path. So, when a new user registers, a folder is created with his name. Inside the folder is also created a file containing a MD5 hashcode of the password.

Every time a user signs, a file is created inside the user's folder containing the data points of the signature. The data that is stored is exactly the data of the parameter points in the table 3.3. The file name follows the format ***`TYPE_SIGNID`***`.signature`, where *TYPE* can be `sign` and `signtry` and *SIGNID* is the signature's identifier. The



```
delta_x           [ [ [ 0, -1 ], [ 3, -2 ], [ 10, -4 ], 20 more... ], [ [ 444, 3 ], [ 455, 6 ],
                  [ 470, 11 ], 119 more... ], [ [ 1368, 0 ], [ 1368, 0 ] ], 4 more... ]
delta_y           [ [ [ 0, 3 ], [ 3, 8 ], [ 10, 14 ], 20 more... ], [ [ 444, -3 ], [ 455, -8 ],
                  [ 470, -14 ], 119 more... ], [ [ 1368, 0 ], [ 1368, 0 ] ], 4 more... ]
signature         [ [ [ 0, 45.91, 15.01 ], [ 1, 44.84, 18.91 ], [ 6, 43.5, 23.83 ], 21 more... ],
                  [ [ 439, -5.4, 54.99 ], [ 449, -2.39, 51.09 ], [ 462, 1.42, 46.17 ], 120
                  more... ], [ [ 1369, 108, 68 ], [ 1368, 108, 68 ], [ 1369, 108, 68 ] ], 4
                  more... ]
log               "Mean Signature\nbSplineDe...e\nbSpline2ndDerivative\n"
speed_x           [ [ [ 0, -2.83 ], [ 1, -3.58 ], [ 6, -4.5 ], 21 more... ], [ [ 439, 8.03 ],
                  [ 449, 10.15 ], [ 462, 12.75 ], 120 more... ], [ [ 1369, 0 ], [ 1368, 0 ],
                  [ 1369, 0 ] ], 4 more... ]
speed_y           [ [ [ 0, 10.39 ], [ 1, 13.14 ], [ 6, 16.5 ], 21 more... ], [ [ 439, -10.39 ],
                  [ 449, -13.14 ], [ 462, -16.5 ], 120 more... ], [ [ 1369, 0 ], [ 1368, 0 ],
                  [ 1369, 0 ] ], 4 more... ]
variation         [ [ [ 0, 2.83 ], [ 3, 6.41 ], [ 10, 10.79 ], 20 more... ], [ [ 444, 76.69 ],
                  [ 455, 75.58 ], [ 470, 74.24 ], 119 more... ], [ [ 1368, 266.66 ], [ 1368,
                  266.66 ] ], 4 more... ]
maxtime           36.0233918128655
position_x        [ [ [ 0, 45.91 ], [ 1, 44.84 ], [ 6, 43.5 ], 21 more... ], [ [ 439, -5.4 ],
                  [ 449, -2.39 ], [ 462, 1.42 ], 120 more... ], [ [ 1369, 108 ], [ 1368, 108 ],
                  [ 1369, 108 ] ], 4 more... ]
position_y        [ [ [ 0, 15.01 ], [ 1, 18.91 ], [ 6, 23.83 ], 21 more... ], [ [ 439, 54.99 ],
                  [ 449, 51.09 ], [ 462, 46.17 ], 120 more... ], [ [ 1369, 68 ], [ 1368, 68 ],
                  [ 1369, 68 ] ], 4 more... ]
acceleration_y    [ [ [ 0, 7.33 ], [ 1, 9.17 ], [ 6, 11 ], 21 more... ], [ [ 439, -7.33 ], [ 449,
                  -9.17 ], [ 462, -11 ], 120 more... ], [ [ 1369, 0 ], [ 1368, 0 ], [ 1369, 0 ] ],
                  4 more... ]
acceleration_x    [ [ [ 0, -2 ], [ 1, -2.5 ], [ 6, -3 ], 21 more... ], [ [ 439, 5.67 ], [ 449,
                  7.08 ], [ 462, 8.5 ], 120 more... ], [ [ 1369, 0 ], [ 1368, 0 ], [ 1369, 0 ] ],
                  4 more... ]
maxdist           21.74591786034282
info              Object { drawingtime=3276, pointspersec="156.59", nsegments=7, more... }
  drawingtime     3276
  pointspersec    "156.59"
  nsegments       7
  npoints         513
  time            4641
```

Figure 3.13: Example of a JSON object sent by the server with information about a signature

signatures with type `sign` are used to create the reference signature of the user and the signatures with type `signtry` are the signatures gathered either from the login as from when the user to checks the signature.

The user defines his signature during the enrollment process and, later, he can define it again many times he wants. All these signatures are kept in the database. When is necessary to get the signatures to create the reference signature, the signatures with type `sign` are sorted by date and only the most recent are used for that.

Additionally, we also create an auxiliary file on the system's path for each request to a new signature identifier. These files create the relation between the signature identifiers and who is the owner of the signature. The file contains information about the time when the request was made, the owner, the signature identifier, the type of signature and an optional info. This optional info can be used to know from where it came from, if it was from the login of if it was checking the signature (the own user or a forger). In the next example shows a file of a request that was made in `2015-06-22` at `16:16:05` by the user `enok`, it have the ID `kl97dpdwbs9ws6g5kx3` and is was an attempt verify the signature (`SIGNATURE_TRY`) in the `login` page:

```
2015-06-22_16:16:05
enok
kl97dpdwbs9ws6g5kx3
SIGNATURE_TRY
login
```

The efficiency and scalability of this approach depends on the file-system where the storage is. This can have implications as on the number of users and signatures as on the performance of the system.

The operating system used was a Linux distribution and with that an ext4 file-system was used. According to the specification of ext4, the file system has no limit on the number of items per folder [37]. For file lookup ext4 implements a HTree which have low lookup times by computing a hash of the path, thereby helping to improve performance in large directories [16].

This was the strategy adopted in the beginning of the work due to its simplicity, but it also revealed to be effective to store the information. During the development of the work did not arise the need to change to another model. In later developments possibly it will be necessary to change to a relational database when the performance will be essential or when the complexity require it.

As an improvement to the storage of the system, it should implement a standard for handwritten signatures. This will allow to easily use datasets in this format and make tests with that. It will also allow easily publish the dataset gathered by us. The standard that defines the format of a handwritten signature is the ISO 19794-7 [26].

Chapter 4

Testing and Results

Due to time constraints, we did not have the opportunity to fully test the system. We intend to make use of the methods as described in section *2.4 System optimization* in the state of the art, that will allow to adjust the system to perform better. But, for now, we only have the tests made previously as described in [21]. But even so we will discuss about some important results obtained from there.

The system was previously tested by 16 people and each one enrolled and checked their signature. After, 4 people tried to forge the signatures by looking at them, i.e. a skilled forgery.

During the calculation of the reference signature, we calculate the average maximum distance. We start by finding the maximum distance between all points used to obtain the resulting point of the reference signature and then we calculate the average distance of all distances. That value is used as threshold for distance in the parameters of comparison. To improve the results, we also applied different multipliers to the average maximum distance. It increases (or decreases if smaller than one) the room of maneuver in comparison with reference signature.

Table 4.1 shows the variation of the multiplier and how the system performed. It is sorted by the column EER and the value represented is $FRR - FAR$. Then, if we search for the value that is closest to zero, it will be the point where the FAR and FRR are close to each other. The column *Distance Multiplier* is the multiplier used and *Author Accept*, *Author Reject*, *Forger Accept* and *Forger Reject* are the counting of signatures with the respective description. The best multiplier found was 1.7 because it was the one that had the EER value closer to 0.

Distance Multiplier	Author Accept	Author Reject	Forger Accept	Forger Reject	FRR	FAR	EER
0.0	0	28	0	41	1.0000	0.0000	1.00000
0.7	1	27	0	41	0.9643	0.0000	0.96429
1.0	5	23	1	40	0.8214	0.0244	0.79704
1.5	18	10	8	33	0.3571	0.1951	0.16202
1.6	18	10	12	29	0.3571	0.2927	0.06446
1.7	19	9	13	28	0.3214	0.3171	0.00436
1.8	22	6	15	26	0.2143	0.3659	-0.1516
1.9	23	5	15	26	0.1786	0.3659	-0.1873
2.0	23	5	16	25	0.1786	0.3902	-0.2117
2.5	25	3	22	19	0.1071	0.5366	-0.4294
3.0	27	1	23	18	0.0357	0.5610	-0.5253

Table 4.1: Variation of the multiplier for distance

Then, we have introduced time in the comparison and a similar method was used as for distance. When we are calculating the reference signature, we find which is the maximum difference in time between the points used to obtain the point in the reference signature. Then, we can get the value for the average maximum difference time. That value is used as threshold for time in the parameters of comparison. Like for distance, to improve the results, we applied different multipliers to the average time, making the time window larger.

Table 4.2 shows the different performances of the system when two parameters for time and for distance were varied. Like the previous table is sorted by EER. So some values are missing because otherwise the table would be extensively long. The best values found were 3.7 for time and 2.6 for distance.

The main finding with this study was time have a relevant impact in the accuracy of the system, whose offline signature verification can not use such information to help in the comparison. Using time, we were able to make the system around 10% more accurate, by improving the results for false positives and the false negatives. This system was 78.26% accurate.

These results are still valid in the current state of the application. What we have done was to add more parameters that can be controlled in the comparison to make it better. But at this point of the work, we still did not performed analytic analysis

Time Mult	Distanc. Mult	Author Accept	Author Reject	Forger Accept	Forger Reject	FRR	FAR	EER
0.5	0.5	0	28	0	41	1.0000	0.0000	1.00000
1.0	1.0	1	27	0	41	0.9643	0.0000	0.96429
2.0	1.0	4	24	0	41	0.8571	0.0000	0.85714
1.0	2.0	7	21	0	41	0.7500	0.0000	0.75000
1.5	4.0	12	16	3	38	0.5714	0.0732	0.49826
2.0	2.0	14	14	1	40	0.5000	0.0244	0.47561
4.5	1.5	17	11	4	37	0.3929	0.0976	0.29530
2.5	3.5	17	11	5	36	0.3929	0.1220	0.27091
3.5	2.5	20	8	9	32	0.2857	0.2195	0.06620
3.0	4.0	18	10	12	29	0.3571	0.2927	0.06446
4.0	2.5	21	7	9	32	0.2500	0.2195	0.03049
3.4	3.3	20	8	11	30	0.2857	0.2683	0.01742
3.4	3.4	20	8	11	30	0.2857	0.2683	0.01742
3.5	2.9	21	7	10	31	0.2500	0.2439	0.00610
3.6	2.9	21	7	10	31	0.2500	0.2439	0.00610
3.6	3.0	21	7	10	31	0.2500	0.2439	0.00610
3.6	3.1	21	7	10	31	0.2500	0.2439	0.00610
3.7	2.6	22	6	9	32	0.2143	0.2195	-0.0052
3.7	2.7	22	6	9	32	0.2143	0.2195	-0.0052
3.7	2.8	22	6	9	32	0.2143	0.2195	-0.0052
3.8	2.6	22	6	9	32	0.2143	0.2195	-0.0052
3.8	2.7	22	6	9	32	0.2143	0.2195	-0.0052
3.8	2.8	22	6	9	32	0.2143	0.2195	-0.0052
3.9	2.6	22	6	9	32	0.2143	0.2195	-0.0052
3.9	2.7	22	6	9	32	0.2143	0.2195	-0.0052
3.9	2.8	22	6	9	32	0.2143	0.2195	-0.0052
3.2	3.7	20	8	12	29	0.2857	0.2927	-0.0070
4.0	3.0	22	6	10	31	0.2143	0.2439	-0.0296
4.5	2.5	23	5	9	32	0.1786	0.2195	-0.0409
3.5	3.5	21	7	12	29	0.2500	0.2927	-0.0427
4.0	3.5	22	6	12	29	0.2143	0.2927	-0.0784
4.5	3.0	24	4	10	31	0.1429	0.2439	-0.1011
4.0	4.0	22	6	13	28	0.2143	0.3171	-0.1028
4.5	3.5	24	4	12	29	0.1429	0.2927	-0.1498

Table 4.2: Variation of the multiplier for time and distance

with the new parameters for velocity and acceleration.

A precise and rigorous comparison between our system and systems presented in the state of the art is impossible because would be necessary to have the same testing sets and those are not publicly available. But comparing the performances obtained with our system in these tests with the referenced systems, our system performed quite well. According with referenced systems in section *2.5 Comparison of signature verification systems* the average accuracy is around 90% and our system is not very far from that. We still have room to make good improvements in our system allowing to achieve an accuracy of that order. It is also important to refer that we did not use pressure and pen inclination to achieve these results, which is not the case of those systems.

We have collected some datasets and we plan to make further testing with them. This will allow to configure the parameters to their optimal values. Those optimal values are the values described in the section *2.2.2 Biometric system performance* and the strategy that we will adopt to determine the values is described in the section *2.4 System optimization*. The datasets that we have collected are:

- **Collected Users** - is the dataset collected by us from real persons, with authentic and forged signatures. We have collected position and time
- **SigComp2015 training dataset** - comprises German, Italian, and Bengali signatures, and English handwritten text. It contains parameters for position and pressure [34].
- **SigWiComp2009 dataset** - evaluation and training data that was used for the SigWiComp2009 competition, not the complete dataset due writers permission. The parameters used are position, pressure, pen tilt and azimuth [8]
- **ATVS-SSig DB** - two datasets of fully on-line synthetic signatures generated and the parameters used are position, timestamp, penups and pressure:
  - DATASET 1 (DS1_Modification_TimeFunctions) the duplicated samples are generated modifying directly the time functions of the master signature. It contains 25 signatures of 350 users, the first 5 signatures follow an intra-session variability and the remaining 20 an inter-session variability.
  - DATASET 2 (DS2_Modification_LNparameters) the duplicated samples are generated modifying the LogNormal parameters of the master signature. It contains 25 signatures of 350 users, the first 5 signatures follow an intra-session variability and the remaining 20 an inter-session variability [54]

4.1 Performance analysis

In this section we will make some tests to infer system performance. We tested the performance of the database and the performance of the signature comparison.

The tests were performed in a computer with the following specs:

- **Processor**: Intel® Core™ i7-2670QM CPU @ 2.20GHz × 8
- **Memory**: 6GB DDR3
- **Disk**: Kingston V300 120GB SSD
- **Operating System**: Kernel Linux 4.1.6-1-ARCH x86_64

4.1.1 Database storage performance

A test was performed to determine if the database is capable of handling a significant number of requests. We made a simple test, but representative of a real case which was to create 100.000 folders. The shell command used was:

```
for i in {1..100000}; do mkdir $i; done
```

We performed this test in several file-systems to understand what is the impact. We selected 4 file-systems: ReiserFS because of the performance features, Ext4 and Ext3 because of the native support on Linux systems and NTFS because of its popularity. Then, we measured the time that took to create and list the folders. Table 4.3 shows the measured times. The file-system that performed better creating the folders was Ext3, but regarding to listing was ReiserFS.

Despite of these results, we still are using an Ext4 file-system for this prototype because is more convenient.

4.1.2 Signature comparison performance

To determine the performance of the system we performed two tests, one to the construction of the reference signature and the other to the signature comparison.

File-system	**Create**			**List**		
	real	user	sys	real	user	sys
ReiserFS	2m32.93s	0m18.17s	0m38.99s	0m00.64s	0m0.21s	0m00.42s
Ext4	3m04.40s	0m21.59s	1m00.14s	0m01.27s	0m0.29s	0m00.77s
Ext3	1m57.36s	0m17.08s	0m53.26s	0m01.08s	0m0.33s	0m00.59s
NTFS	3m14.43s	0m21.22s	1m56.93s	0m05.05s	0m0.38s	0m01.70s

Table 4.3: Time to create and list 100.000 folders

Time (ms) – (723 users)		**Stats**	
2747	2110	Avg	2194.8
2177	2126	Std	186.65
2124	2104	Max	2747
2115	2204	Min	2104
2105	2136	Median	2125
Time per user: 3.04ms			

Table 4.4: Performance test for building reference signature

#	**Reference(ms)**	**Get Signs.(ms)**	**Comparing(ms)**	**Total(ms)**
1	2260	6388	34940	43590
2	2260	5640	33292	41194
3	2175	5539	31959	39675
4	2177	5488	32591	40256
5	2219	5637	33357	41214
6	2223	5583	33237	41043
7	2220	5750	33489	41460
8	2243	5786	33768	41797
9	2246	5846	33713	41805
10	2222	5679	33241	41142
Avg	2224.5	5733.6	33358.7	41317.6
Std	28.44	242.04	734.31	980.25
Max	2260	6388	34940	43590
Min	2175	5488	31959	39675
Median	2222.5	5659.5	33324.5	41204
Per each	(user) 3.08	(user) 7.93	(compar.) 2.35	(sign.) 5.42

Table 4.5: Performance test for comparing signatures

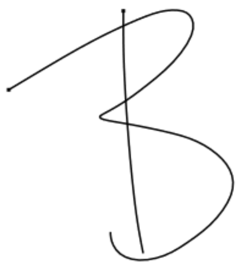
Starting by testing the construction of the reference signature, we measured the time that took to calculate the mean signature. The experiment was performed 10 times with all users registered in the database, 723 in total. Table 4.4 shows the time that each trial took as well as the statistics associated. Based on the average value, we can say the construction of the reference signature take 3.04ms to accomplish.

Regarding to the comparison of signatures, it is more complex than the reference signature. Before the comparison take place, it is necessary for each user, build the reference signature and get a list of their signatures. The way the system was built, we have to create the reference signature every time we want to compare a signature because we still do not store those calculations. Then, we can understand how much time we will reduce if we store the reference signature.

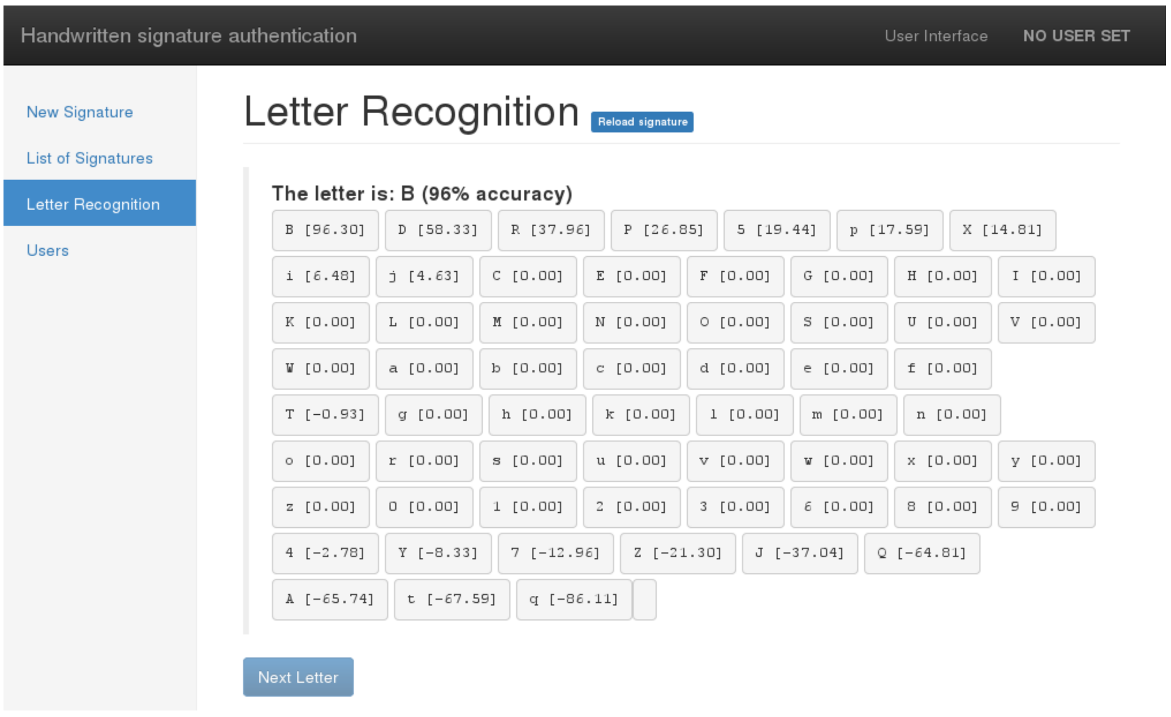
The tests were made 10 times with all 723 users making a total of 14 221 signature comparisons. Table 4.5 shows the time that this test took as well as the statistics associated. Then we calculated the time that took each comparison. The values regarding to building the reference signature and getting the list of signatures are per user, 3.08ms and 7.93ms respectively. Only the comparison of two signatures took 2.35ms on average, but in reality, when we want to compare a signature we have first to build the reference signature, so the total time to compare a signature is 5.42ms ($reference\ signature + comparison$). The time to get the list of signatures is not taken into account because is not necessary when comparing a single signature. Theoretically, a system of this dimension is capable of handling around 180 comparisons per second or 11 000 per minute. Meanwhile, there is still room for many improvements by optimizing the procedures. These tests are just about the time spent with computations, it does not include server/client response times.

4.2 Letter recognition

A interesting application that was achieved easily was the recognition of handwritten uppercase and lowercase letters and numbers. The idea is to create a database with all letters and numbers using the same system that we have been describing in this work. The same idea can be also extended for words, but it begins not to be so feasible. Then to recognize, the users writes the letter or the number instead of the signature and the system will compare that with all records in the database. The most similar match is the correspondent character. This idea is substantially different from an OCR system.



(a) The letter written



Letter Recognition

The letter is: B (96% accuracy)

```
B [96.30]  D [58.33]  R [37.96]  P [26.85]  5 [19.44]  p [17.59]  X [14.81]
i [6.48]  j [4.63]  C [0.00]  E [0.00]  F [0.00]  G [0.00]  H [0.00]  I [0.00]
K [0.00]  L [0.00]  M [0.00]  N [0.00]  O [0.00]  S [0.00]  U [0.00]  V [0.00]
W [0.00]  a [0.00]  b [0.00]  c [0.00]  d [0.00]  e [0.00]  f [0.00]
T [-0.93]  g [0.00]  h [0.00]  k [0.00]  l [0.00]  m [0.00]  n [0.00]
o [0.00]  r [0.00]  s [0.00]  u [0.00]  v [0.00]  w [0.00]  x [0.00]  y [0.00]
z [0.00]  0 [0.00]  1 [0.00]  2 [0.00]  3 [0.00]  6 [0.00]  8 [0.00]  9 [0.00]
4 [-2.78]  Y [-8.33]  7 [-12.96]  Z [-21.30]  J [-37.04]  Q [-64.81]
A [-65.74]  t [-67.59]  q [-86.11]
```

(b) Result of the recognition

Figure 4.1: Example of the recognition of the letter B

Figure 4.1 illustrates the recognition of a letter. Actually the recognition was very effective with the tested set, the hardest case detected was between `V` and `U` because these letters are very similar.

This application can be used to unlock the phone. Instead of the user have to draw for example the lock pattern, he can write a letter to unlock the phone. This will possibly make the lock code more secure because it have in consideration the time that took to make the letter, which does not happen with lock pattern. Most of the times, the lock pattern is guessable by looking to the stains on the screen made by the fingers.

This functionality is only available in the administrative interface.

Chapter 5

Conclusions and Future Work

We are currently assisting a revolution in the authorization systems. Example of this is growing integration of biometric sensors in smartphones and integrated authentication solutions. There is also research being done and published in recognition of handwritten signatures, making this an active field of study. These factors make of this work actual and with high relevance in the context that we live today. The continuous growth of the internet and the increasing security requirements for the development of the e-society, the field of automatic signature verification is being considered with renewed interest since it uses an already accepted authentication method at legal and social levels [25].

The use of mobile devices is a promising way to disseminate the use of handwritten signatures in the digital world. We are certain that this is an important step in the way to a more modern society, more ecological and more efficient. In the age of e-society is very important the emergence of applications of automatic signature verification to solve problems of our daily life and do not allow a tool like this to be kept only as a research academic topic.

5.1 Research summary

During this work we studied how to create a system capable of recognizing handwritten signatures and use that as a method of authentication. This imposed since from the beginning several challenges. We had to create a simple but innovative application by using recent technologies. This system can be used by users to be authenticated with their handwritten signature in a convenient and cost effective way. We had investigated a way to collect signatures with the current technologies.

With this work we described a fully working authentication system using handwritten signatures, enhanced with mobile devices and provided with innovative solutions to collect handwritten signatures from the users. We have also described and taken inspiration from several other proposed handwritten signature recognition systems, because they serve to illustrate and to help understand how to approach the problem of verifying signatures and to identify the weaknesses of other and making this work better.

We believe that we have accomplished the main objectives that we had proposed to achieve in the beginning of the work:

- We researched about the main topics of handwritten biometric authentication
- We made a definition of an architecture of an authentication mechanism based on handwritten signatures
- We have implemented the proposed system
- We deployed the system in real scenarios to analysis the performance and usage
- We made a proof of concept that is possible to use mobile devices to recognize users and provide some authenticity based on biometric handwritten signatures

However, the current state of the application is very much a work in progress. What has been done until now can only be seen as a first step for a more complete proof of concept that now must be exercised by being evaluated in large scale scenarios, enabling the dissemination of the results with its stakeholders and encourage them to use this system as a means of authentication.

5.2 Main findings

We can infer from the current state of the art that signature verification and new ways of authentication are nowadays a very hot research field with a high interest not only due to the sheer number of authors that dedicate their research budgets over it, but also by the the real impact they can have on society and the way of millions of users will interact with digital documents and services provided by the Internet in the near future. However, we have also found that none of the systems we have studied make authenticity of a digital document based on handwritten signatures, except the platform SIGN*ificant* by XYZMO (`https://www.xyzmo.com/`) which is similar to the system that we proposed. Nevertheless, they provided us mechanisms and guidelines about how the system should work.

QRCodes are effectively an easy and convenient way to exchange information between a computer and a mobile device. They allow quickly exchange complex data that would very hard for humans to replicate. HTML Canvas also fit perfectly in this task, it provides the required functionality, are distributed with every mobile device and are independent of platform.

Mobile devices proved to be an excellent option to be used to collect the signatures from the users because they are widely available and are good enough to make the readings of biometric features. However we don't know how the absence of extra parameters, like pressure and pen inclination, can affect the results.

Using the time dimension in the verification process is an important factor. In this case of the proposed system, the improvement was around 10%. The accuracy of the system was 78.26% meanwhile, according with the studied systems in the state of the art, the average accuracy was around 90%. Our system is not very far from that and we believe further refining of the process will help to achieve better results.

5.3 Current implementation limitations

The system we have developed still have some limitations that we want to address in the future work. The system is already fully operational, but it is more a work in progress than a system ready to be deployed, due to the complexity of the subject.

One current limitation that affects directly the usage of the application is the requirement of the signatures from the same user of having the same number of segments. We had users that experienced some problems because they made small variations in the signature. We also have to adapt the algorithm to be tolerant to points that failed the comparison, otherwise a signature may become difficult to be accepted.

More tests should be performed in order to adjust the parameters of the system to a optimal (or at least as good as possible) state. This requires an exhaustive testing in various conditions and with an extensive database of signatures.

The database can impose also some limitations by affecting the responsiveness of the system. That can be caused by high lookup time for a file or by deadlocks due to multiple access. A relational database could be a better solution.

5.4 Future work

This proposed method requires further development and testing should me made to be used on real applications. We already discussed some points that should be improved in the previous section *5.3 Current implementation limitations*, like the restriction on the number of segments, tolerance to failed points, further testing and a transition to relational database.

Regarding to testing, we pretend to enlarge the number of users by publishing the system with CRACS/INESC members, who are potentially around 750. Not only signers should contribute, but also with forgers to test a wider variety of situations, closer as possible to reality. Other variations and parameters of the proposed system should be consider a determine which one is better. Usability tests should also be performed to help to understand better if the system is easily usable.

New approaches to verify the signatures should also be considered. In the state of the art we discussed different approaches used in the literature with handwritten signature verification and some should be tested to evaluate which is best. The signatures gathered during the enrollment process can have a bigger role in the comparison, for example they can be used to identify different variations in the signatures that users make when signing.

To attest the effectiveness of these proposed improvements, the participation in handwritten signature competitions should be considered. Another test that can be made is to test the system with devices that are able to acquire pressure and pen inclination and compare the gains achieved in terms of accuracy relatively to time/positioning. Thus we will understand better how these extra parameters could affect the result and what are the real practical improvements that could be obtained with a larger set of biometric features. Usually the competitions use pressure in the test sets.

One major improvement that must be taken into account is the security. The system is currently vulnerable attacks like man-in-middle attack and theft and replication of signatures. To solve that the communications should be secured and should be adopted strategies to secure the client application to not get compromised. The integration of digital certificates associated with handwritten signatures will also help to make this system more secure.

We have planed the development of an Alfresco (`https://www.alfresco.com/`) extension that will allow in the integration of this system into the document workflow. It will allow a document be signed by a group of people that are required to sign that document in a secure and efficient way. This extension is already under development. Another possible future integration is passwordless UX as a way of authentication in the framework developed by FIDO Alliance (`https://fidoalliance.org/`).

We also intend to use the material which have been developed to make publications in conferences. Some of them are ICB2016 - International Conference on Biometrics (`http://icb2016.hh.se`), ICIAP2016 - International Conference on Image Analysis and Processing (`https://www.waset.org/conference/2016/10/paris/ICIAP`) and ICFHR2016 - International Conference on Frontiers in Handwriting Recognition (`http://www.nlpr.ia.ac.cn/icfhr2016/index.htm`).

5.5 Conclusion

Nowadays does not yet exists a well-accepted solution to this problem and this constitutes an active area of research where it is still possible to make very meaningful contributions and new applications came up everyday.

The proposed system compares signatures using the Euclidean Distance. With that is possible to verify handwritten signatures with some degree of confidence, by just comparing distances. It is possible to obtain meaningful authentication results that tell us whether it is the legitimate user which is signing or if is someone trying to impersonate someone else. However if a user is specialized in forging signatures, he will probably be able to make a successful forgery. Are still many the improvements and tests that should made to make this system ready to be used in real situations.

Handwritten signature verification is an important step for the general massification of digital legal documents. The main reason being the use of handwritten signatures in documents still is, and will be in the foreseeable future, the culturally dominating accepted form of authenticating them.

Appendix A

Development notes

A.1 Software used

- **Open Java Development Kit (JDK)** is a free and open source implementation of the Java Platform, Standard Edition (Java SE).
  - Website: `http://openjdk.java.net/`
  - Version: 1.8.0_31
  - Download link: `https://jdk8.java.net/download.html`
- **Apache Tomcat** is an open-source web server and servlet container capable to run Java Servlets, JavaServer Pages (JSP) and more.
  - Website: `http://tomcat.apache.org/`
  - Version: 8.0.20
  - Download link: `http://mirrors.fe.up.pt/pub/apache/tomcat/tomcat-8/v8.0.20/bin/apache-tomcat-8.0.20.tar.gz`
- **Java JSON** is a library that provides support to encode/decode JSON in Java and includes the capability to convert between JSON and other formats.
  - Website: `http://www.json.org/java/index.html`
  - Version: 2010-12-24
  - Download link: `http://www.java2s.com/Code/JarDownload/java-json/java-json.jar.zip`

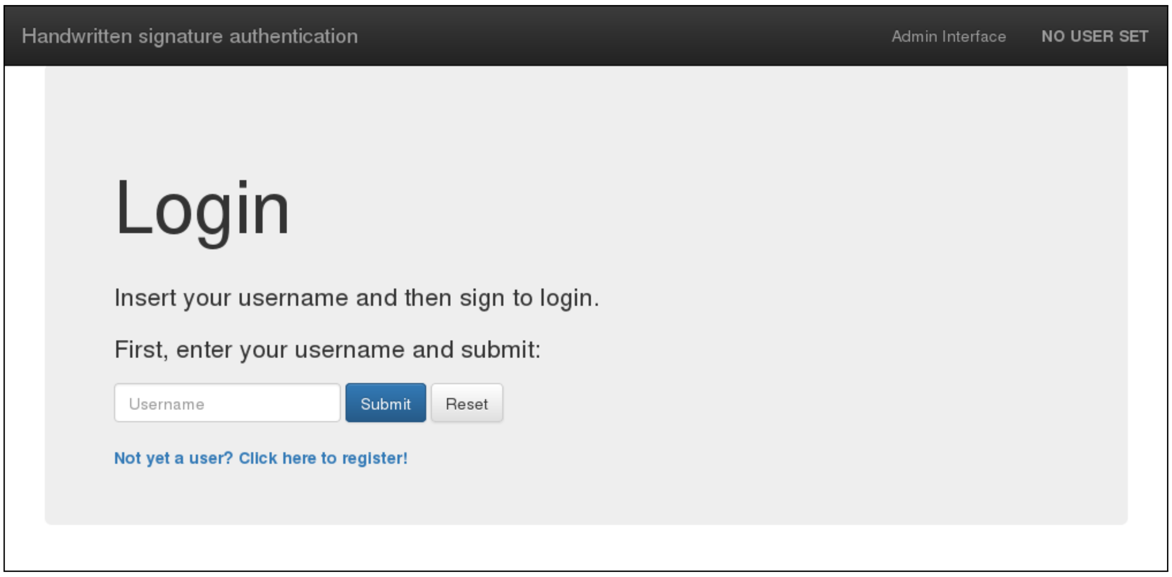
- **Google reCAPTCHA Java Library** is a Java library to help to protect web sites from bots, by displaying a challenge with distorted images.
  - Website: `https://developers.google.com/recaptcha/old/docs/java`
  - Version: 0.0.7
  - Download link: `https://recaptcha.googlecode.com/files/recaptcha4j-0.0.7.zip`
- **JQuery** is a JavaScript library designed to make the web development easier. It is extensible through plug-ins.
  - Website: `https://jquery.com/`
  - Version: 1.11.1
  - Download link: `http://code.jquery.com/jquery-1.11.1.min.js`
- **QRCode** is a JavaScript library intended to create QRCodes and place it into a web page.
  - Website: `http://davidshimjs.github.io/qrcodejs/`
  - Version: 2014-12-12
  - Download link: `https://github.com/davidshimjs/qrcodejs/tarball/master`
- **jQuery Plotting Plugin** is a jQuery JavaScript plug-in to plot graphs and charts.
  - Website: `http://www.jqplot.com/`
  - Version: 1.0.8r1250
  - Download link: `https://bitbucket.org/cleonello/jqplot/downloads/jquery.jqplot.1.0.8r1250.tar.gz`
- **Bootstrap** is a set of tools to help to create websites, by making interfaces richer and with consistent user experience.
  - Website: `http://getbootstrap.com/`
  - Version: 3.3.1
  - Download link: `https://github.com/twbs/bootstrap/releases/download/v3.3.1/bootstrap-3.3.1-dist.zip`

Appendix B

User manual

B.1 Enrollment

1. Access to the website `http://handwrite.hltsys.pt`
   Then click on **Not yet a user? Click here to register!**


Handwritten signature authentication
Admin Interface
NO USER SET

Login

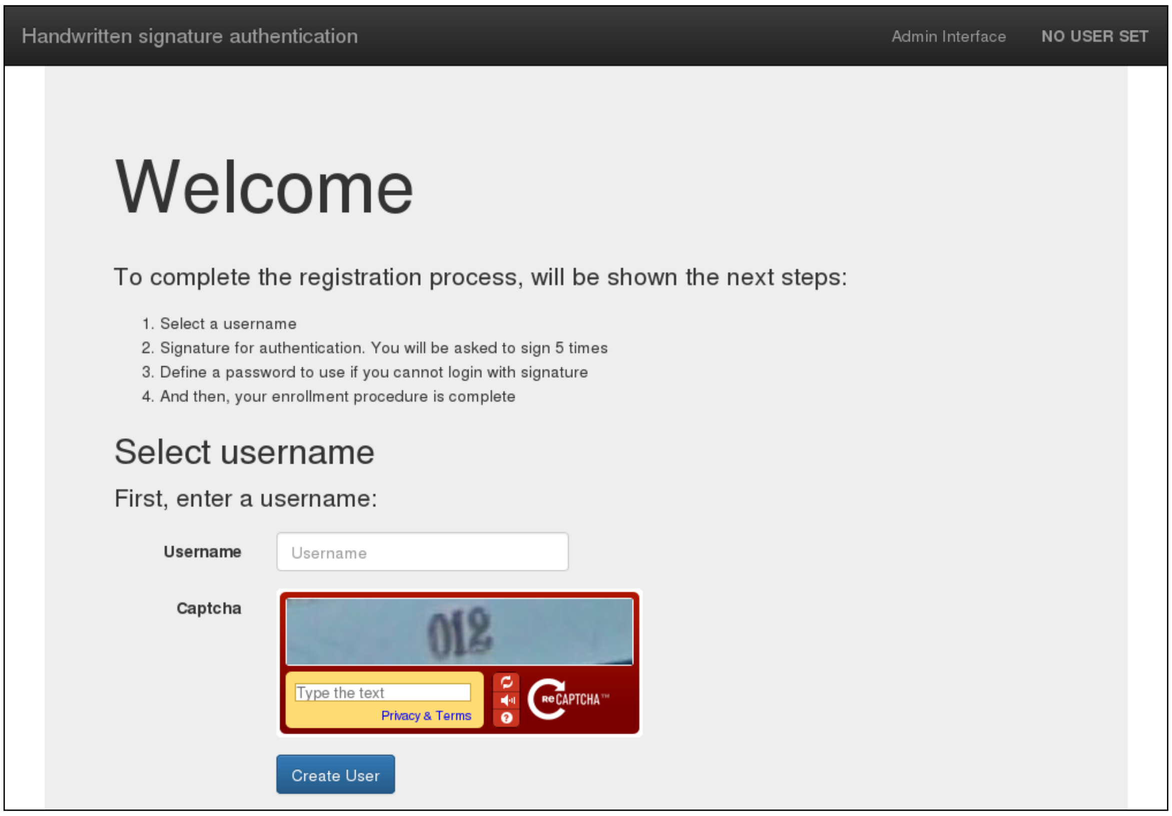
Insert your username and then sign to login.

First, enter your username and submit:

Username
Submit
Reset
Not yet a user? Click here to register!

Figure B.1: Starting page

2. Select a username. The username you are creating should not exist previously. You also have to confirm you are a person by inputting a captcha.
   Press **Create User**.


Handwritten signature authentication

Welcome

To complete the registration process, will be shown the next steps:

1. Select a username
2. Signature for authentication. You will be asked to sign 5 times
3. Define a password to use if you cannot login with signature
4. And then, your enrollment procedure is complete

Select username

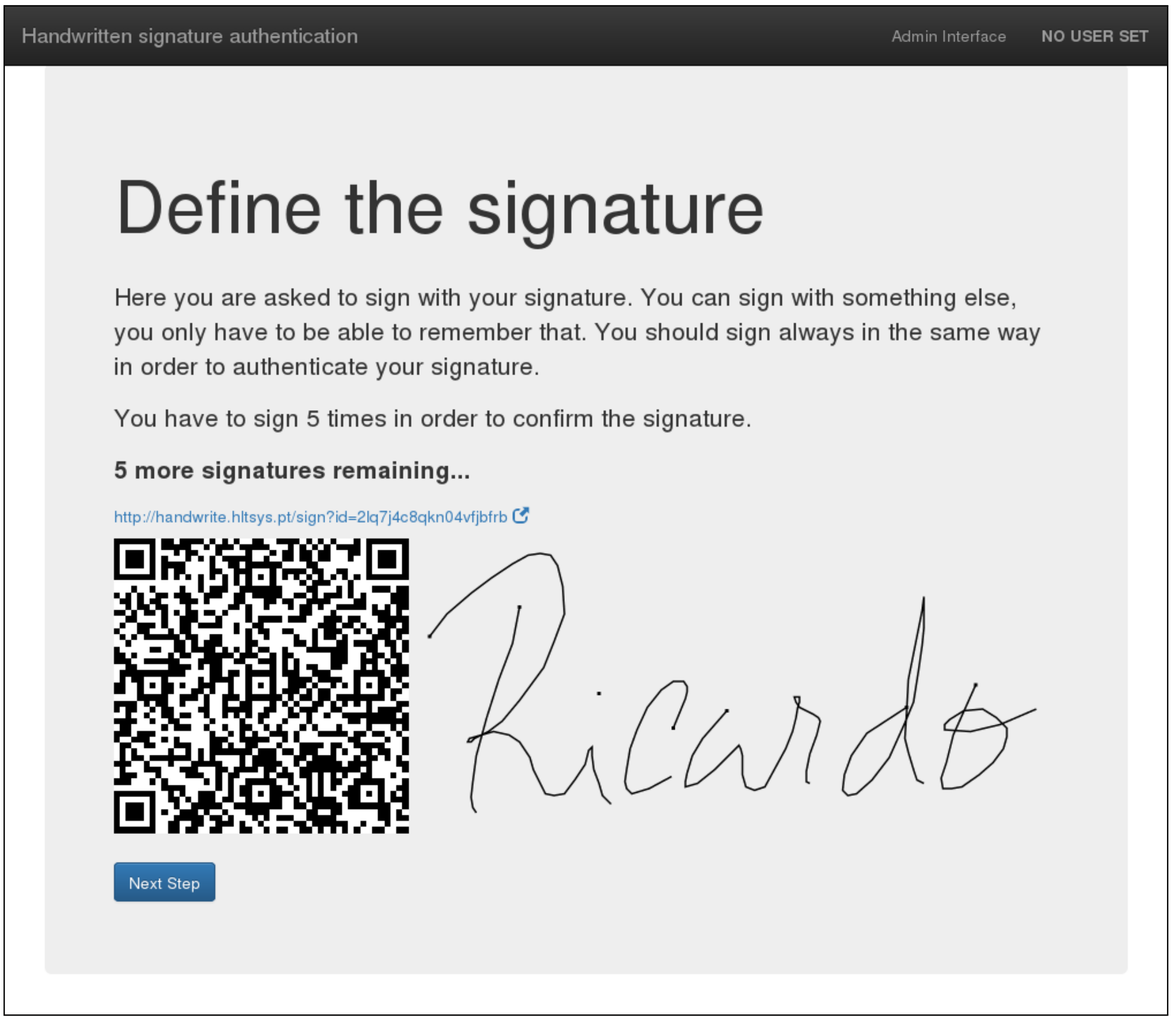
First, enter a username:

Username: Username

Captcha

Figure B.2: Welcome page to new users

3. Now read the QRCode with your smartphone which contains a URL. Open it with a browser. There is a counter indicating how many signatures left.



Handwritten signature authentication

Admin Interface NO USER SET

Define the signature

Here you are asked to sign with your signature. You can sign with something else, you only have to be able to remember that. You should sign always in the same way in order to authenticate your signature.

You have to sign 5 times in order to confirm the signature.

5 more signatures remaining...

http://handwrite.hltsys.pt/sign?id=2lq7j4c8qkn04vfjbfrb

Ricardo

Figure B.3: Signature acquisition during enrollment

4. This is the application that will be opened. Here you can sign your signature, you are be asked to sign 5 times.
   You can use your finger or stylus for better accuracy.

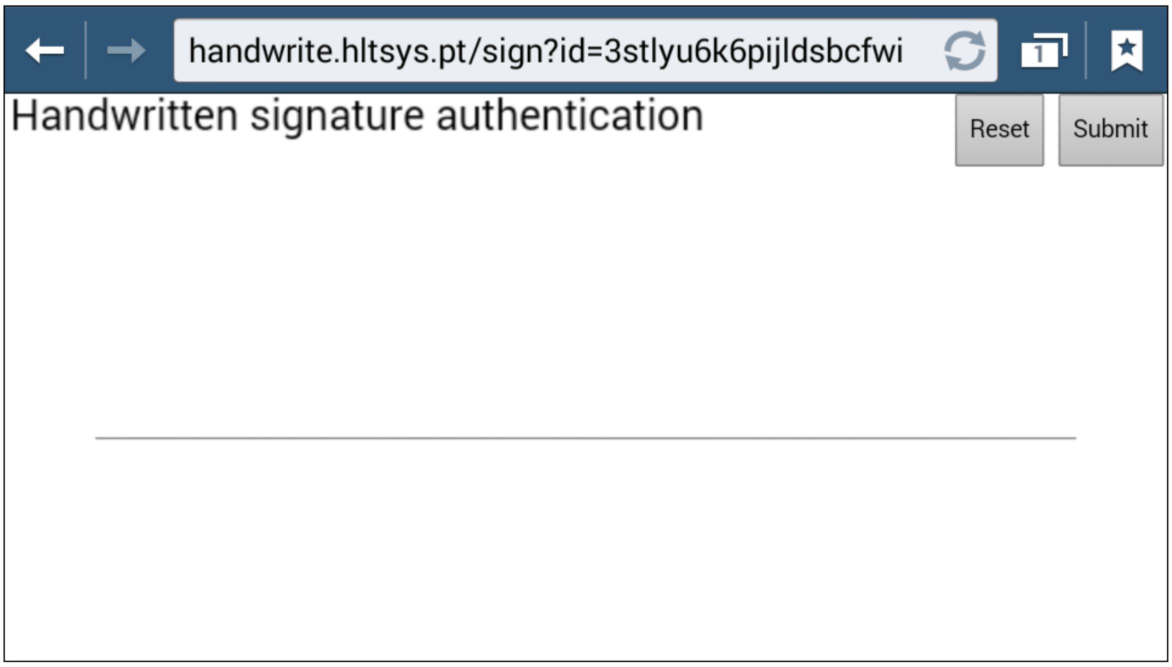
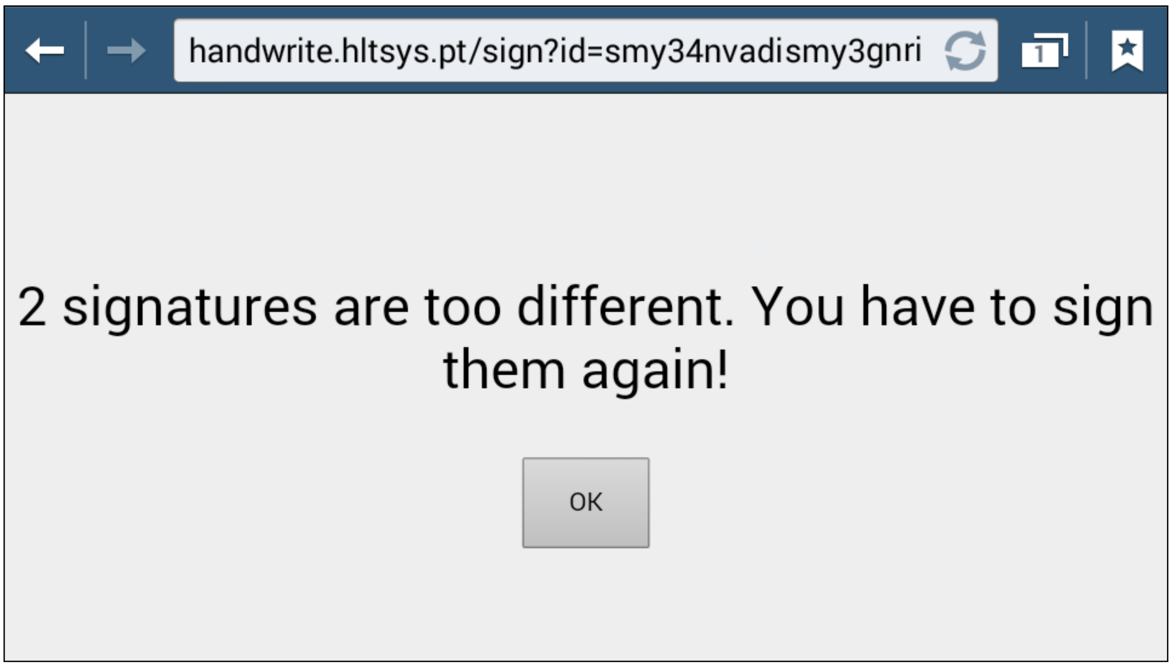

handwrite.hltsys.pt/sign?id=3stlyu6k6pijldsbcfwi
Handwritten signature authentication
Reset
Submit

Figure B.4: Web application to acquire signatures on smartphone

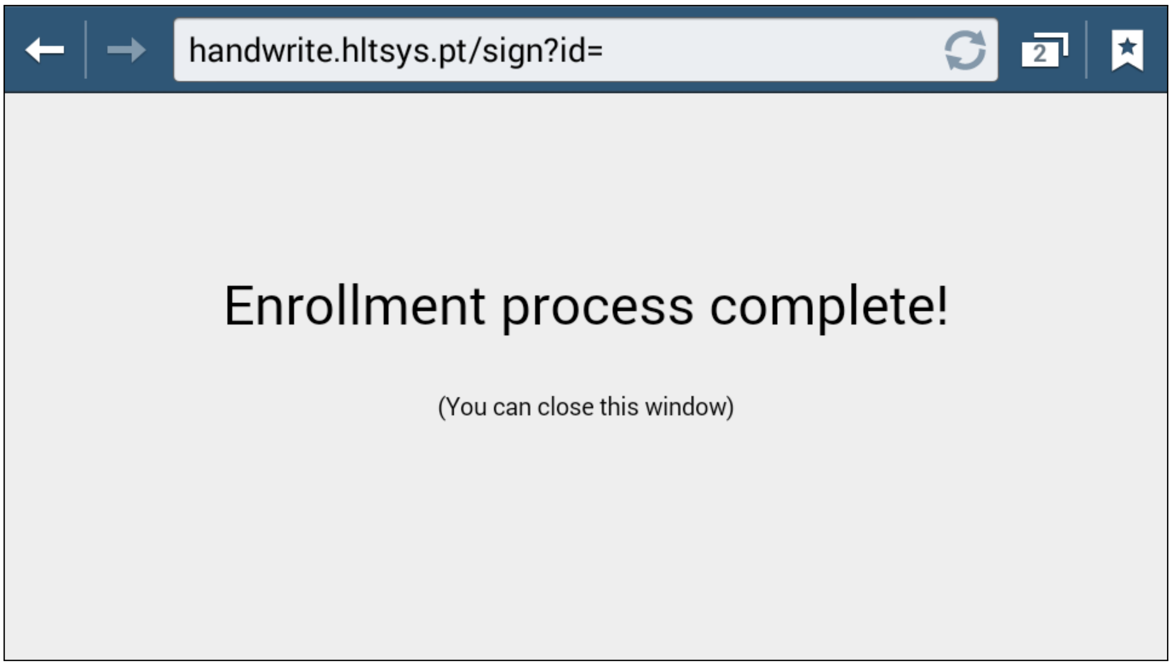
5. If some of the signatures are too different from the others, the system will ask you to sign them again.
   Press **OK** to sign the remaining signatures.


handwrite.hltsys.pt/sign?id=smy34nvadismy3gnri

2 signatures are too different. You have to sign them again!

Figure B.5: Error during enrollment process

6. When all signatures are similar, you will be presented with a message like this.

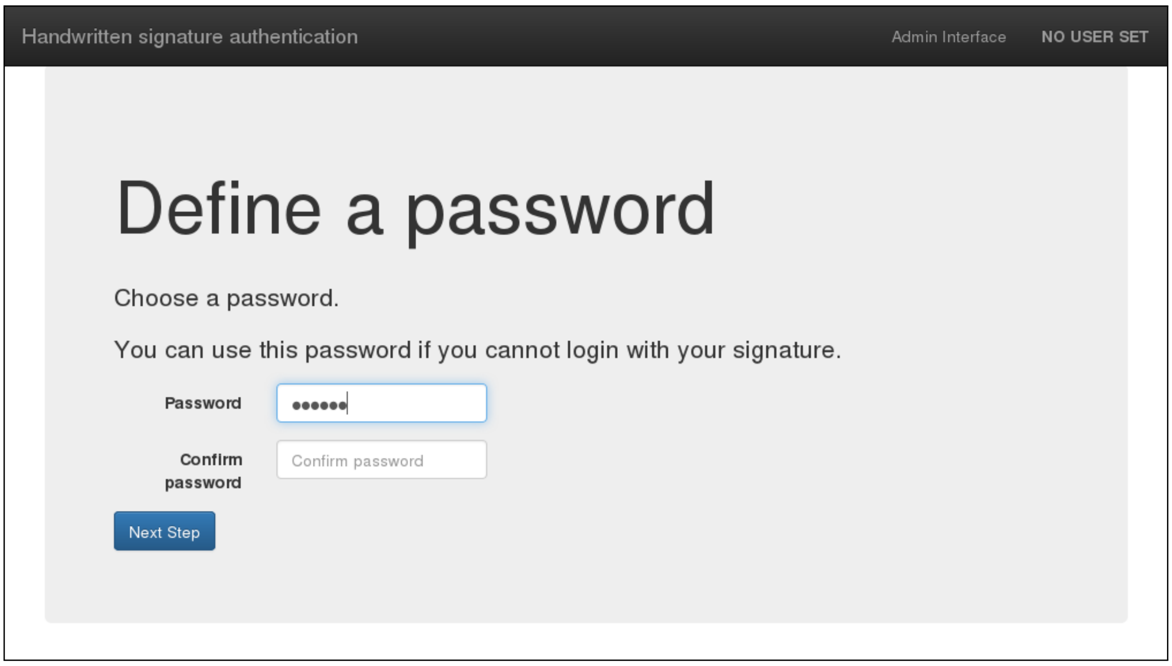

handwrite.hltsys.pt/sign?id=

Enrollment process complete!

(You can close this window)

Figure B.6: Message displayed when all signatures are similar

7. Then, you are asked to create a password. This can be used if you have problems to login with your signature. The password should contain at least 6 characters. Click **Next Step** to proceed.



Define a password

Choose a password.

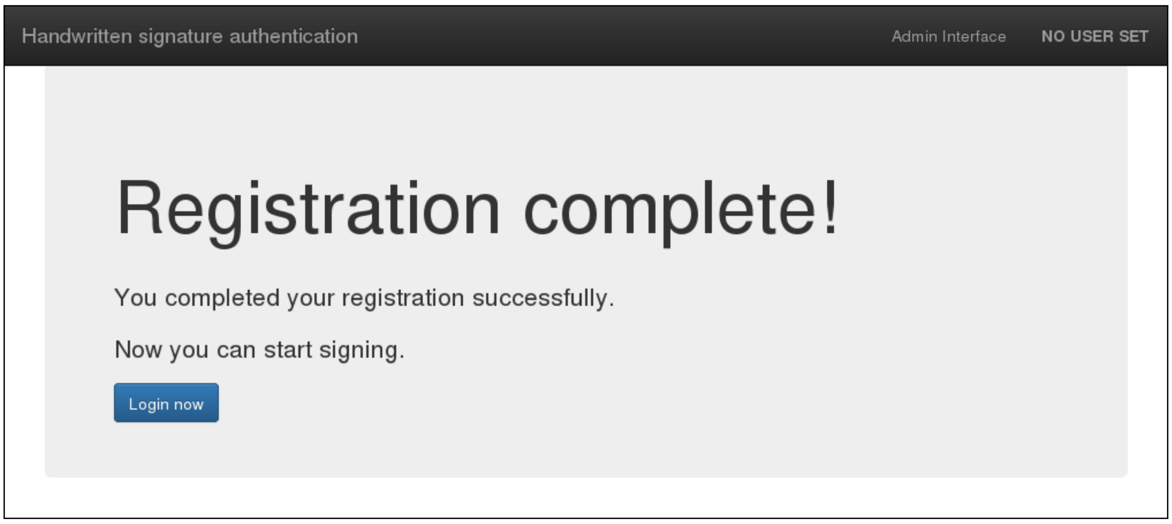
You can use this password if you cannot login with your signature.

Password ••••••

Confirm password Confirm password

Figure B.7: Defining a password to use as recovery

8. **Congratulations!** The enrollment process is now complete.
   Press **Login now** to login into your account.



Registration complete!

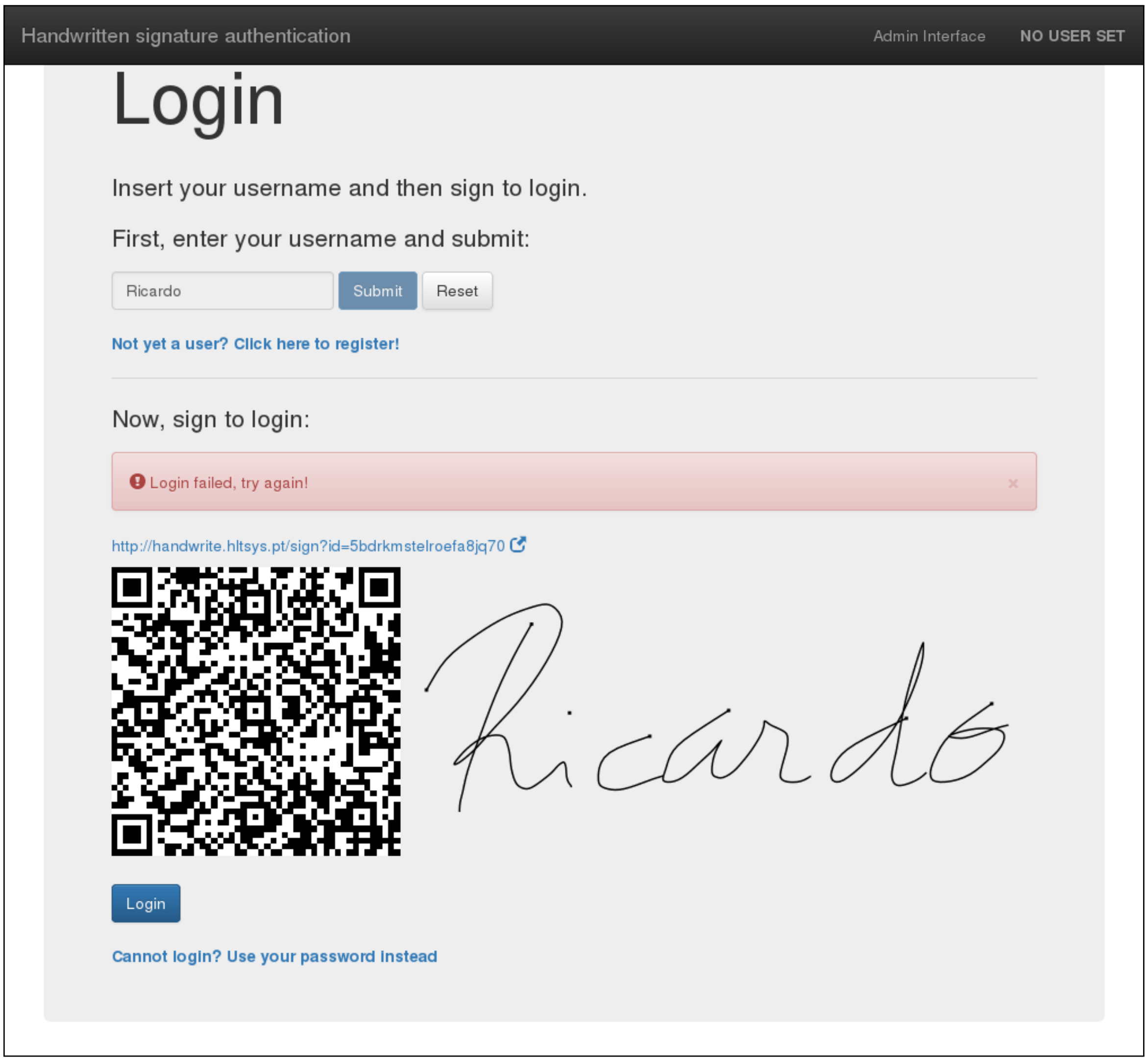
You completed your registration successfully.

Now you can start signing.

Figure B.8: Enrollment process completion

B.2 User account

1. Access to the website `http://handwrite.hltsys.pt`
   Insert your username in the text field and press **Submit**
   Then use your smartphone to read the QRCode. Similarly to the enrollment process, open the URL and sign.
   If your signature is not recognized, you can sign again. When accepted, a message will be displayed on the smartphone.



Login

Insert your username and then sign to login.

First, enter your username and submit:

Ricardo

Not yet a user? Click here to register!

Now, sign to login:

Login failed, try again!

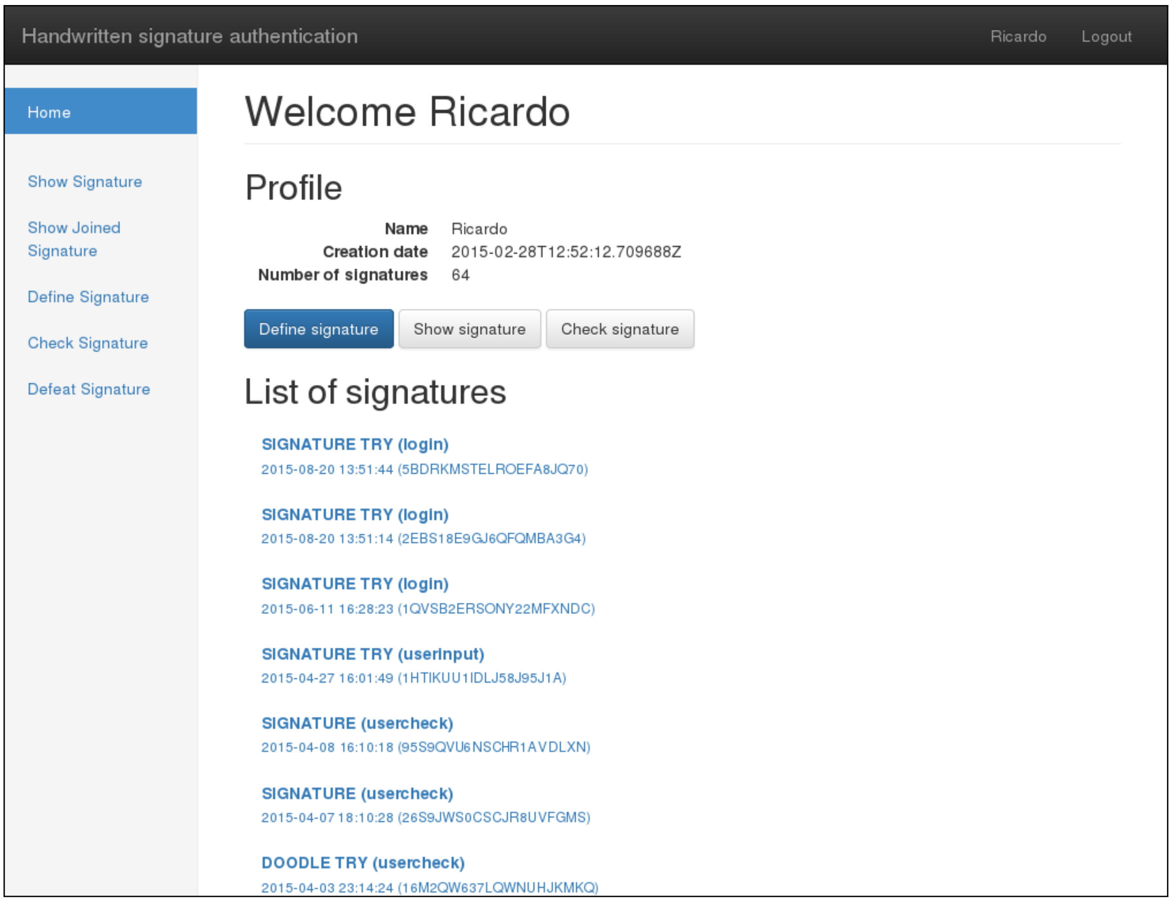
http://handwrite.hltsys.pt/sign?id=5bdrkmstelroefa8jq70

Ricardo

Cannot login? Use your password instead

Figure B.9: Login page

2. Account home page.
   From the left menu, the user can access to the functionalities of the system. In the center, the user can see some info about him and a list of signatures all associated with him.



Welcome Ricardo

Profile

Name	Ricardo
Creation date	2015-02-28T12:52:12.709688Z
Number of signatures	64

List of signatures

SIGNATURE TRY (login)
2015-08-20 13:51:44 (5BDRKMSTELROEFA8JQ70)

SIGNATURE TRY (login)
2015-08-20 13:51:14 (2EBS18E9GJ6QFQMBA3G4)

SIGNATURE TRY (login)
2015-06-11 16:28:23 (1QVSB2ERSONY22MFXNDC)

SIGNATURE TRY (userinput)
2015-04-27 16:01:49 (1HTIKUU1IDLJ58J95J1A)

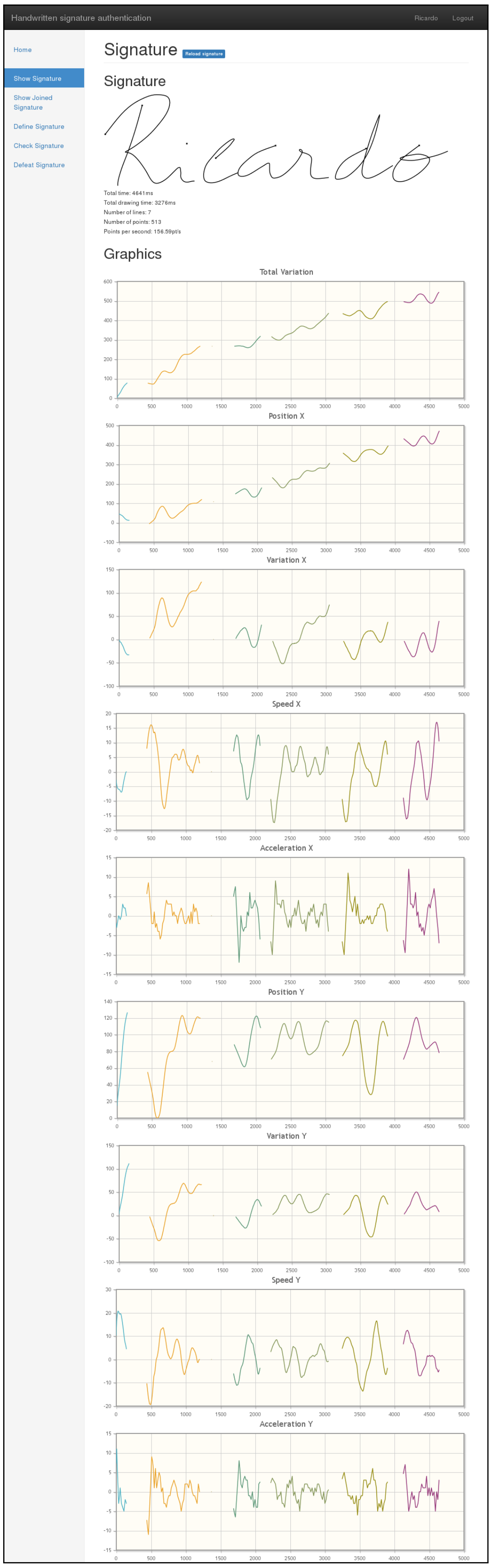
SIGNATURE (usercheck)
2015-04-08 16:10:18 (95S9QVU6NSCHR1AVDLXN)

SIGNATURE (usercheck)
2015-04-07 18:10:28 (26S9JWS0CSCJR8UVFGMS)

DOODLE TRY (usercheck)
2015-04-03 23:14:24 (16M2QW637LQWNUHJKMKQ)

Figure B.10: Account home page

3. After selecting **Show Signature**, the system shows which is your reference signature. There is displayed detailed information about the signature, like position, variation, speed and acceleration.



Signature

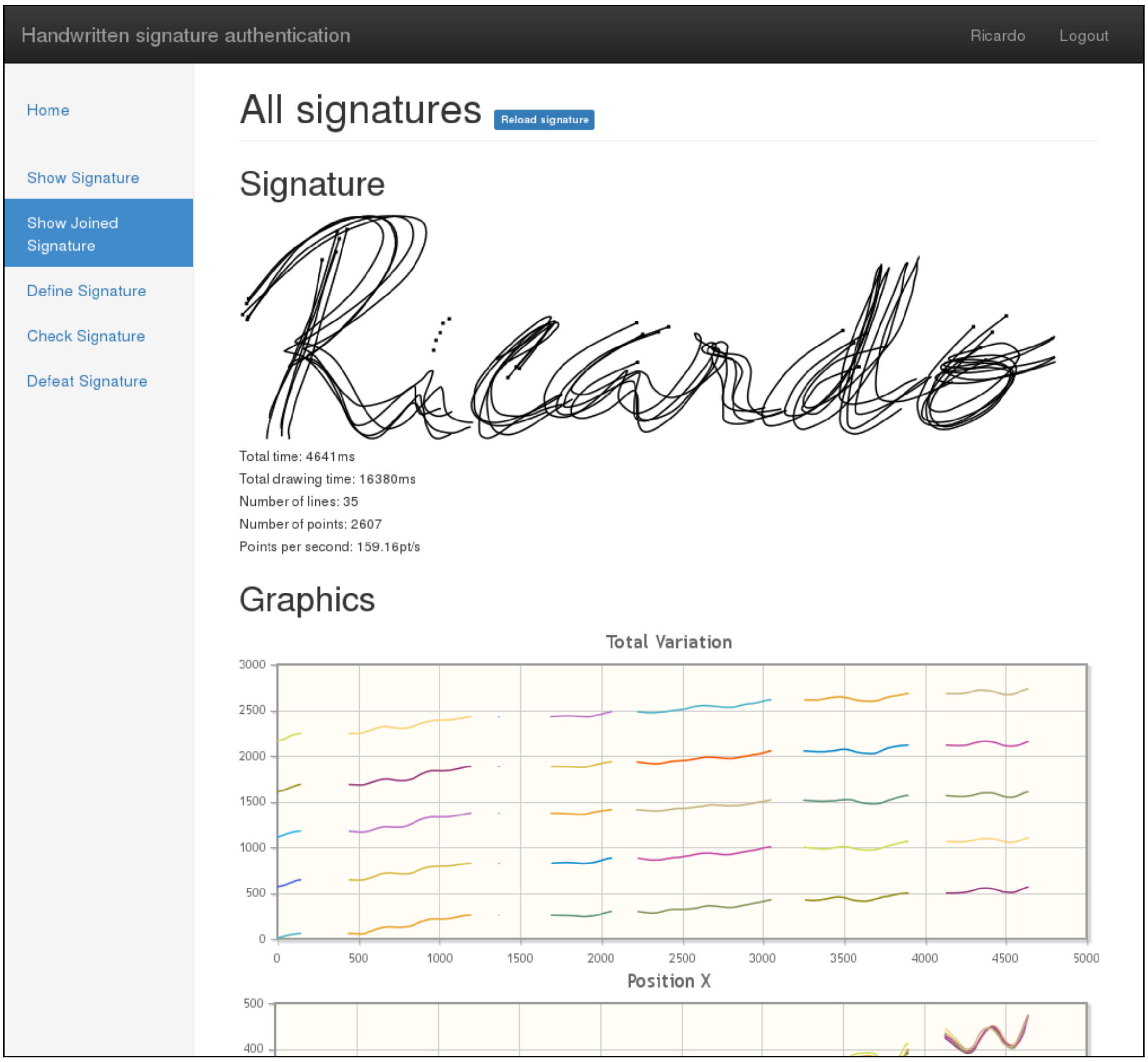
Signature

Total time: 4641ms
Total drawing time: 3276ms
Number of lines: 7
Number of points: 513
Points per second: 156.59pt/s

Graphics

Figure B.11: Detailed information about a signature

4. In the option **Show Joined Signature** is possible to see signatures used to build the reference signature.
   The signatures will appear overlapped to be easier to compare them. The detailed information will be shown also together.



Total time: 4641ms
Total drawing time: 16380ms
Number of lines: 35
Number of points: 2607
Points per second: 159.16pt/s

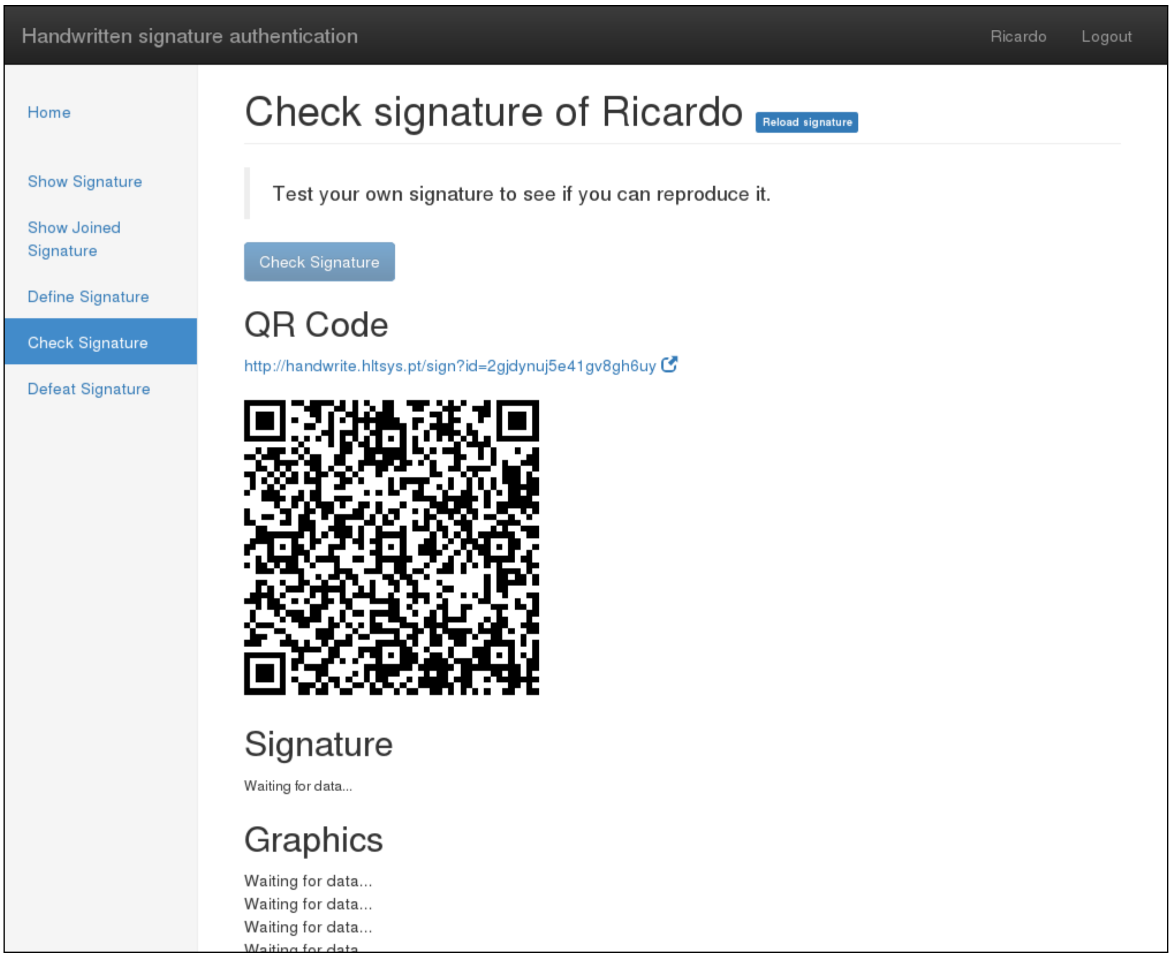
Figure B.12: All signatures together

5. By selecting **Define Signature**, you can redefine your signature.
   You can use this option if you're having trouble logging on.

6. The options **Check Signature** and **Defeat Signature** are very similar.
   Here you can sign and compare your signature with the signature registered on the system.

   - **Check Signature** - intended to be used by you, the author of the signature.

   - **Defeat Signature** - intended to be used by someone else that you want to invite to try to replicate your signature.



Check signature of Ricardo

Test your own signature to see if you can reproduce it.

QR Code

http://handwrite.hltsys.pt/sign?id=2gjdynuj5e41gv8gh6uy

Signature

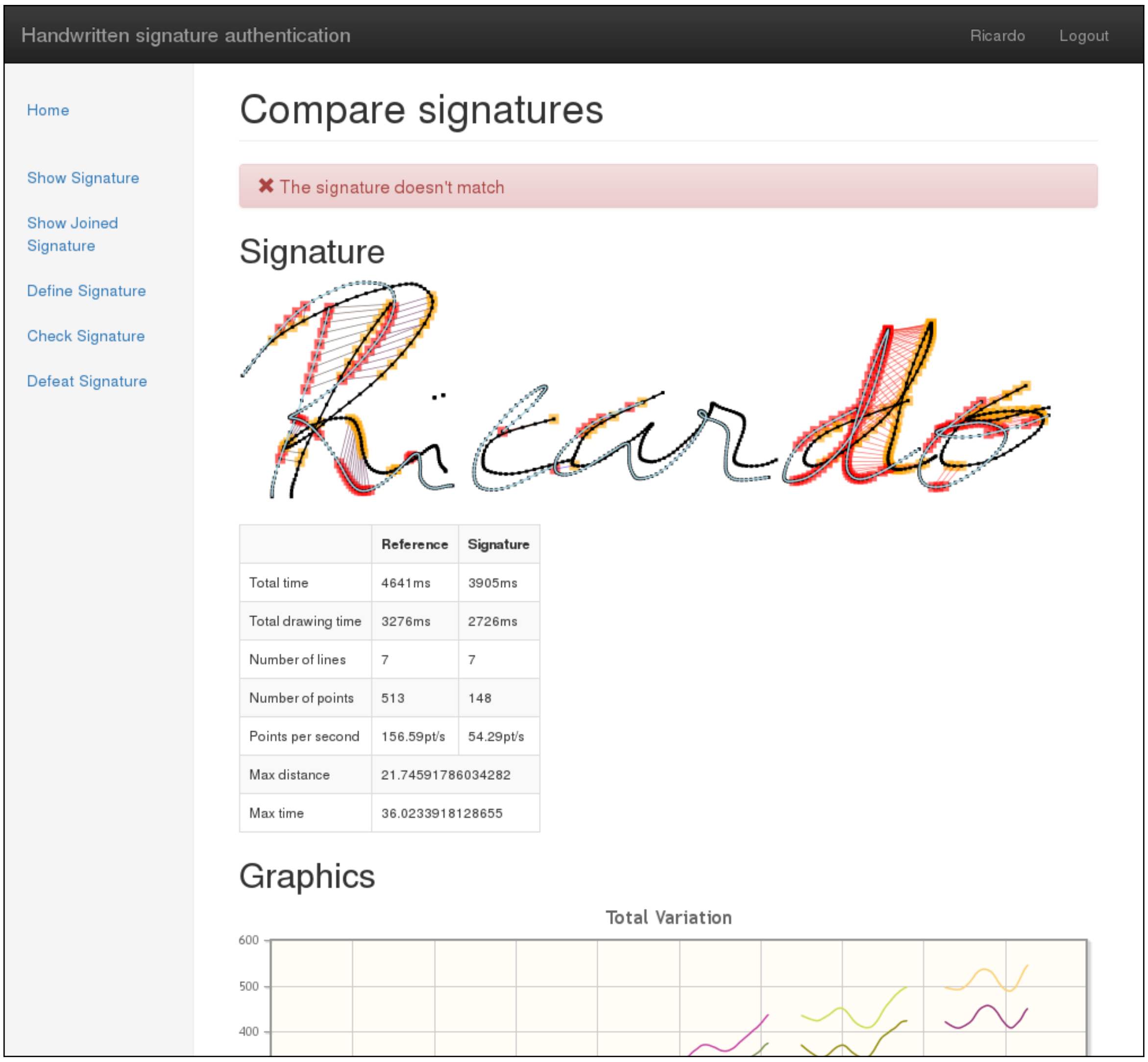
Waiting for data...

Graphics

Waiting for data...
Waiting for data...
Waiting for data...

Figure B.13: Checking signature

7. From the account home page is possible to select each signature from the list that is associated with you. Then, you can see the detailed information about that signature.
   If you press the button **Compare signature**, the selected signature is compared with your reference signature.
   The signature in black is the selected signature and the signature in light blue is the reference signature. The points where the comparison failed are represented in red in the reference signature and in orange the correspondent point in the selected signature. A textual description about the comparison can be accessed by clicking in the result.



Compare signatures

✖ The signature doesn't match

Signature

	Reference	Signature
Total time	4641ms	3905ms
Total drawing time	3276ms	2726ms
Number of lines	7	7
Number of points	513	148
Points per second	156.59pt/s	54.29pt/s
Max distance	21.74591786034282	
Max time	36.0233918128655	

Graphics

Figure B.14: Comparison between the signature and the reference

References

[1] B-spline basis functions: Definition. `http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/B-spline/bspline-basis.html`. [Online; accessed 10-March-2015].

[2] A brief history of signatures. `http://electronicsignature.com/a-brief-history-of-signatures`, August 2011. [Online; accessed 02-September-2015].

[3] AFBarstow, teoli, dtapuska, jpmedly, fscholz, Jeremie, momoi, ksvrouvas, shubham.hatwar, Sheppy, wesj, and dflanagan. Touch. `https://developer.mozilla.org/en-US/docs/Web/API/Touch`, 2015. [Online; accessed 21-September-2015].

[4] Fahad Aijaz, Muzzamil Aziz Chaudhary, and Bernhard Walke. Performance comparison of a soap and rest mobile web server. *Third International Conference on Open-Source Systems and Technologies (ICOSST 2009)*, 2009.

[5] Julio Angulo and Erik Wästlund. Exploring touch-screen biometrics for user identification on smart phones. In *Privacy and Identity Management for Life*, pages 130–143. Springer, 2012.

[6] Sevinc Bayram, Husrev Taha Sencar, and Nasir Memon. A survey of copy-move forgery detection techniques. In *IEEE Western New York Image Processing Workshop*, pages 538–542. Citeseer, 2008.

[7] Vivian L Blankers, CEVD Heuvel, Katrin Y Franke, and Louis G Vuurpijl. Icdar 2009 signature verification competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1403–1407. IEEE, 2009.

[8] V.L. Blankers, C.E. van den Heuvel, K. Franke, and L. Vuurpijl (et al). The icdar 2009 signature verification competition. In *Proceedings of the 2009 10th*

International Conference on Document Analysis and Recognition, pages 1403–1407, 2009.

[9] Stephen E. Blythe. Fine-tuning vietnam's electronic transactions law to promote growth in e-commerce. *Journal of International Commercial Law and Technology*, 9(4):214–228, 2014.

[10] Franco Callegati, Walter Cerroni, and Marco Ramilli. Man-in-the-middle attack to the https protocol. *IEEE Security & Privacy*, (1):78–81, 2009.

[11] Anu Chaudhary, Girija Srikanth, and Pawan Bhadana. Fast offline signature verification using qr code on printed cheques: A review. *International Journal for Advance Research in Enguneering and Technology*, vol. 2(no. 3), March 2014.

[12] Pria Chetty. An analysis of electronic signature regulation in south africa. Master's thesis, Faculty of Management, University of the Witwatersrand, March 2013.

[13] Eileen Y. Chou. Paperless and soulless: E-signatures diminish the signer's presence and decrease acceptance. *Social Psychological and Personality Science*, pages 1–9, December 2014.

[14] Adobe Document Cloud. esign service, e-signatures online, formerly echosign. `https://acrobat.adobe.com/us/en/documents/esignatures.html`. [Online; accessed 13-September-2015].

[15] DocuSign. How electronic signature (esignature) works. `https://www.docusign.com/how-it-works/electronic-signature`. [Online; accessed 13-September-2015].

[16] Kevin D. Fairbanks. An analysis of ext4 for digital forensics. *Digital Investigation*, Volume 9, Supplement:S118–S130, August 2012.

[17] S. Fauziyah, O. Azlina, B. Mardiana, A.M. Zahariah, and Hazura Haroon. Signature verification system using support vector machine. *6th International Symposium on Mechatronics and its Applications, 2009. ISMA '09*, pages 1–4, March 2009.

[18] William J. Flynn. Electronic signature forensics. `http://topazsystems.com/Information/forensics.pdf`, 2015. [Online; accessed 02-September-2015].

[19] fscholz, teoli, MHassan, Pettay, Khannabyss, kscarfone, Sheppy, acusti, bigbossSNK, and wesj. Touchevent. `https://developer.mozilla.org/en-US/docs/Web/API/TouchEvent`, 2014. [Online; accessed 30-October-2014].

[20] Steve Fulton and Jeff Fulton. *HTML5 canvas.* O'Reilly Media, Inc., 2013.

[21] Ricardo P. Gonçalves, Alexandre B. Augusto, and Manuel E. Correia. Time/space based biometric handwritten signature verification. *10th Iberian Conference on Information Systems and Technologies (CISTI), 2015*, vol. 2:pp.1–6, June 2015.

[22] Christian Gruber, Christian Hook, Jürgen Kempf, Georg Scharfenberg, and Bernhard Sick. A flexible architecture for online signature verification based on a novel biometric pen. *Adaptive and Learning Systems, 2006 IEEE Mountain Workshop on*, pages 110–115, July 2006.

[23] Juan J Igarza, Iñaki Goirizelaia, Koldo Espinosa, Inmaculada Hernáez, Raúl Méndez, and Jon Sánchez. Online handwritten signature verification using hidden markov models. In *Progress in Pattern Recognition, Speech and Image Analysis*, pages 391–399. Springer, 2003.

[24] D. Impedovo, G. Pirlo, and R. Plamondon. Handwritten signature verification: New advancements and open issues. *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on. IEEE*, pages 367–372, September 2012.

[25] Donato Impedovo and Giuseppe Pirlo. Automatic signature verification: The state of the art. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 38(no. 5):609–635, September 2008.

[26] ISO. Information technology - biometric data interchange formats - part 7: Signature/sign time series data. ISO ISO/IEC 19794-7: 2014, International Organization for Standardization, Geneva, Switzerland, 2014.

[27] Ramanujan S Kashi, Jianying Hu, WL Nelson, and William Turin. On-line handwritten signature verification using hidden markov model features. In *Document Analysis and Recognition, 1997., Proceedings of the Fourth International Conference on*, volume 1, pages 253–257. IEEE, 1997.

[28] Marzuki Khalid, Hamam Mokayed, Rubiyah Yusof, and Osamu Ono. Online signature verification with neural networks classifier and fuzzy inference. *Third Asia International Conference on Modelling & Simulation, 2009. AMS '09*, pages 236–241, May 2009.

[29] M. Khalid Khan, M. Aurangzeb Khan, Mohammad A. U. Khan, and Imran Ahmad. On-line signature verification by exploiting inter-feature dependencies. *18th International Conference on Pattern Recognition, 2006. ICPR 2006*, vol.2:796–799, August 2006.

[30] lambert. B splines. `http://www.cse.unsw.edu.au/~lambert/splines/Bspline.html`. [Online; accessed 10-March-2015].

[31] M. Liwicki, C.E. van den Heuvel, B. Found, and M.I. Malik. Forensic signature verification competition 4nsigcomp2010 - detection of simulated and disguised signatures. In *Frontiers in Handwriting Recognition (ICFHR), 2010 International Conference on*, pages 715–720, Nov 2010.

[32] Marcus Liwicki, Muhammad Imran Malik, Linda Alewijnse, Elly van den Heuvel, and Bryan Found. Icfhr 2012 competition on automatic forensic signature verification (4nsigcomp 2012). In *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*, pages 823–828. IEEE, 2012.

[33] Marcus Liwicki, Muhammad Imran Malik, C Elisa van den Heuvel, Xiaohong Chen, Charles Berger, Reinoud Stoel, Michael Blumenstein, and Bryan Found. Signature verification competition for online and offline skilled forgeries (sigcomp2011). In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1480–1484. IEEE, 2011.

[34] Muhammad Imran Malik, Marcus Liwicki, Sheraz Ahmed, Angelo Marcelli, Michael Blumenstein, Umapada Pal, Srikanta Pal, and Linda Alewijnse. Sigwicomp2015 - signature verification and writer identification & retrieval competitions. In *Proceedings of the 13 th International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

[35] Muhammad Imran Malik, Marcus Liwicki, Linda Alewijnse, Wataru Ohyama, Michael Blumenstein, and Bryan Found. Icdar 2013 competitions on signature verification and writer identification for on-and offline skilled forgeries (sigwicomp 2013). In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1477–1483. IEEE, 2013.

[36] Davide Maltoni, Dario Maio, Anil K Jain, and Salil Prabhakar. *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.

[37] Avantika Mathur, Mingming Cao, Suparna Bhattacharya, Andreas Dilger, Alex Tomas, and Laurent Vivier. The new ext4 filesystem: current status and future plans. *Linux Symposium*, vol. 2:21–34, June 2007.

[38] Alan McCabe, Jarrod Trevathan, and Wayne Read. Neural network-based handwritten signature verification. *Journal of Computers*, vol. 3(no. 8), August 2008.

[39] Yuxin Meng, Duncan S Wong, Roman Schlegel, et al. Touch gestures based biometric authentication scheme for touchscreen mobile phones. In *Information Security and Cryptology*, pages 331–350. Springer, 2013.

[40] Oscar Miguel-Hurtado, Luis Mengibar-Pozo, Michael M. Lorenz, and Judith Liu-Jimenez. On-line signature verification by dynamic time warping and gaussian mixture models. *41st Annual IEEE International Carnahan Conference on Security Technology, 2007*, pages 23–29, October 2007.

[41] Gavin Mulligan and Denis Gračanin. A comparison of soap and rest implementations of a service based interaction independence middleware framework. *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 1423–1432, December 2009.

[42] Ellen R. Nichols. *Biometrics: Theory, Applications, and Issues.* Biotechnology in Agriculture, Industry and Medicine. Nova Science Publishers, Inc, 2011.

[43] Marc Parizeau and Réjean Plamondon. A comparative analysis of regional correlation, dinamic time warping, and skeletal tree matching for signature verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2(no. 7), July 1990.

[44] Cesare Pautasso, Olaf Zimmermann, and Frank Leymann. Restful web services vs. “big” web services: making the right architectural decision. *WWW ’08 Proceedings of the 17th international conference on World Wide Web*, pages 805–814, 2008.

[45] João Pedro Pedroso. Simple metaheuristics using the simplex algorithm for non-linear programming. In Thomas Stützle, Mauro Birattari, and Holger H. Hoos, editors, *Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics*, volume 4638 of *Lecture Notes in Computer Science*, pages 217–221. Springer Berlin Heidelberg, 2007.

[46] Linda Pesante. Introduction to information security. `https://www.us-cert.gov/sites/default/files/publications/infosecuritybasics.pdf`, 2008. [Online; accessed 12-September-2015].

[47] R. Plamondon and S.N. Srihari. Online and off-line handwriting recognition: a comprehensive survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):63–84, Jan 2000.

[48] M. Jasmine Pemeena Priyadarsini, K. Murugesan, Srinivasa Rao Inbathini, A. Jabeena, and K. Sai Tej. Bank cheque authentication using signature. *International Journal for Advance Research in Enguneering and Technology*, vol. 3(no. 5), May 2013.

[49] Yu Qiao, Xingxing Wang, and Chunjing Xu. Learning mahalanobis distance for dtw based online signature verification. In *Information and Automation (ICIA), 2011 IEEE International Conference on*, pages 333–338, June 2011.

[50] Ben Rogers. Digital signatures in a pdf. Technical report, Adobe, May 2012.

[51] Foxit Software. Docusign and foxit bring esignature to 130 million customers in 150 countries. `https://www.foxitsoftware.com/portuguese/company/press.php?action=view&page=201209052312.html`, September 2012. [Online; accessed 13-September-2015].

[52] Puchong Subpratatsavee, Peerapon Pudtuan, Jirawan Charoensuk, Thanapob Sondee, and Thananchai Vejchasetthanon. The authentication of handwriting signature by using motion detection and qr code. *Information Science and Applications (ICISA), 2014 International Conference on. IEEE*, May 2014.

[53] TechTarget. X.509 certificate definition. `http://searchsecurity.techtarget.com/definition/X509-certificate`, January 2014. [Online; accessed 09-September-2015].

[54] Biometrics Ideal Test. Description of atvs-ssig db. `http://biometrics.idealtest.org/dbDetailForUser.do?id=12`. [Online; accessed 22-September-2015].

[55] Can I use. Canvas (basic support). `http://caniuse.com/#feat=canvas`, 2015. [Online; accessed 21-September-2015].

[56] USMA. Biometrics metrics report v3.0. Usma, U.S. Military Academy (USMA) – West Point, December 2012.

[57] James Wayman, Anil Jain, Davide Maltoni, and Dario Maio. An introduction to biometric authentication systems. In James Wayman, Anil Jain, Davide Maltoni, and Dario Maio, editors, *Biometric Systems*, pages 1–20. Springer London, 2005.

[58] James L. Wayman. Error rate equations for the general biometric system. *Robotics & Automation Magazine, IEEE*, 6(1):35–48, 1999.

[59] Petra Wohlmacher. Digital certificates: a survey of revocation methods. In *Proceedings of the 2000 ACM workshops on Multimedia*, pages 111–114. ACM, 2000.

[60] XYZMO. E-signing for signature pads. `https://www.xyzmo.com/e-signature-products/sign-on-signature-pad`. [Online; accessed 13-September-2015].

[61] L Yang, BK Widjaja, and R Prasad. Application of hidden markov models for signature verification. *Pattern recognition*, 28(2):161–170, 1995.