UNIVERSITY OF PADUA

DEI – Department of Information Engineering

Master Degree in Computer Engineering

A Multimodal and Multi-Algorithmic Architecture for Data Fusion in Biometric Systems

Supervisor:

Chiar.mo Prof. Carlo Ferrari ……………………………………

Candidate:

Andrea Giacometti Matr. 626604 ………………………………………

Academic Year 2012/2013

Alla mia famiglia, che mi ha sempre sostenuto

e non mi ha mai fatto mancare niente

Ai miei amici che mi hanno sempre aiutato

e a modo loro mi hanno fatto pensare ad altro

Ad i miei compagni d'università, con cui ho passato

molte belle giornate spensierate nonostante fossimo al DEI

A me stesso, che nonostante le mille tentazioni di mollare

ed i periodi in cui era impossibile studiare

non ho ceduto e sono andato avanti

Index

Summary 1
1. Introduction 2
  1.1. What Biometric Is 3
  1.2. Advantages 5
  1.3. Disadvantages 6
2. Biometric Systems 9
  2.1. Components of a Biometric system 9
  2.2. Verification Vs Identification 11
  2.3. Measures of Biometrics performance 14
    2.3.1. Errors of a Biometric system 14
    2.3.2. Performance metrics 17
  2.4. Biometric Traits 22
    2.4.1. DNA 23
    2.4.2. Ear 23
    2.4.3. Face 23
    2.4.4. Facial, hand and hand vein infrared thermogram 24
    2.4.5. Fingerprint 24
    2.4.6. Hand and finger geometry 25
    2.4.7. Iris 25
    2.4.8. Keystroke 25
    2.4.9. Retinal scan 26
    2.4.10. Signature 26
    2.4.11. Voice 26
  2.5. Choosing a Biometric trait 27
  2.6. Limitations of biometrics 29
    2.6.1. Accuracy 29
    2.6.2. Scalability 30
    2.6.3. Security and Privacy 30
3. Multiobiometrics 33
  3.1. Advantages 33
  3.2. Multiobiometric Taxonomy 35
  3.3. Levels of fusion 36
    3.3.1. Fusion at Sensor Level 39

3.3.2. Fusion at Feature extraction Level 39
3.3.3. Fusion at Matching Score Level 40
3.3.4. Fusion at Decision Level 43
3.3.5. Fusion at Rank Level 44
3.4. Score normalization 45
3.5. State of the Art 48
3.5.1. Generalized densities 48
3.5.2. Quality measures 51
3.5.3. Weighted Sum Rule 51
3.6. Summary 53
4.1. Face Authentication 56
4.1.1. Face Detection 56
4.1.2. Face Recognition 60
4.2. Fingerprint Recognition 61
4.2.1. Minutiae extraction 61
4.2.2. Minutia matching 62
4.3. Iris Authentication 64
4.3.1. Iris Detection 64
4.3.2. Iris Recognition 65
5.1. So far Summary 68
5.2. Introduction 69
5.3. Devices 70
5.3.1. Iris sensor 70
5.3.2. Fingerprint sensor 71
5.3.3. Face sensor 72
5.4. Implementation 73
5.4.1. Suit Application choice 73
5.5. First Analisys 73
5.6. Black-box execution 76
5.7. Loading Algorithms 77
5.8. Database 82
5.9. ENROLLMENT 83
5.9.1. Face enrollment 87
5.9.2. Fingerprint enrollment 90
5.9.3. Iris enrollment 91

5.9.4. Adjust thresholds 95
5.10. Verification 96
5.10.1. Fusion and normalization 98
5.11. Analyzing algorithms 104
5.11.1. Test 105
5.11.2. Match score distributions 107
6.1. Future developments 111

Index of figures

Figure 1: Examples of Physiological and Behavioral based biometrics systems 4
Figure 2 : An iris scan at the Schiphol Amsterdam Airport 5
Figure 3 : Generic Biometric System components/units 10
Figure 4 : Enrollment phase 11
Figure 5 : Verification phase 12
Figure 6 : Identification phase 13
Figure 7 : Examples of intra-class variations in (a) face recognition (b) signature recognition 15
Figure 8 : Example of inter-class similarity of (a) twins and (b) father and son 16
Figure 9 : An example of Genuine and Impostor distribution scores. 17
Figure 10 : Binary classification confusion matrix 18
Figure 11 : FAR and FRR percentages to vary the threshold 19
Figure 12 : ROC Space. The diagonal line represents a random binary classifier where for every decision it has 50% of probability to be correct or incorrect. 21
Figure 13 : Receiver operating characteristic (ROC) curve for a matcher 22
Figure 14 : Comparison among several biometric modalities 27
Figure 15 : Three iris images more and more blurred from left to right 29
Figure 16 : Biometric System and the nine different points of attack 31
Figure 17 : An example of a multibiometric system in verification mode using palmprint and face traits. 33
Figure 18 : Various sources of information that can be fused in a multibiometric system. In four of the five scenarios (multiple sensors, representations, instances and samples), multiple sources of information are derived from the same biometric trait. In the fifth scenario, information is derived from different biometric traits and such systems are known as multimodal biometric systems. ... 36
Figure 19 : Stream data flow of the authentication process 37
Figure 20 : Example of serial/cascade acquisition mode. Here the next step is performed if necessary, basing on the previous decision 38
Figure 21 : Fusion at Feature Extraction Level in a multibiometric system 40
Figure 22 : Fusion at Match-Score Level in a multibiometric system 41
Figure 23 : Decision tree example 43
Figure 24 : Fusion at Decision Level in a multibiometric system 44
Figure 25 : Summary diagram of possible levels of information fusion 45
Figure 26 : Approaches to information fusion 54
Figure 27 : Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey

rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature 57

Figure 28 : The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location P1 is the sum of the pixels in rectangle A. The value at location P2 is A + B, at location P3 is A + C and at location P4 is A + B + C + D.............................. 58

Figure 29 : The classifier cascade is a sequence of weak AdaBoost trained classifiers................... 59

Figure 30 : Two Haar like features selected on an face image... 60

Figure 31 : (a) Orientation field by Rao’s method (b) Minutiae extracted (c) Alignment of input pattern on template pattern 63

Figure 32 : Iris detection and phase quadrant coding sequence is illustrated by the bit stream shown graphically 65

Figure 33 : Two irises captured with VW and NIR cameras respectively....................................... 66

Figure 34 : An image of VistaFA2 iris/face sensor produced by Vista Imaging Incorporated........... 71

Figure 35 : An image of the FX3000 fingerprint sensor produced by Biometrika........................... 72

Figure 36 : An image of the workstation used to develop the software 72

Figure 37 : Representation of how the final system should work.. 75

Figure 38 : Functional scheme of Enrollment and Verification phases of the system 76

Figure 39 : The Strategy Pattern 77

Figure 40 : The Strategy pattern applied to our case of generalizing the biometric algorithm 78

Figure 41 : Database tables 83

Figure 42 : Software Initial screen 86

Figure 43 : Insert user data screen 86

Figure 44 : Boundaries of detected face and eyes locations 90

Figure 45 : (a) Live capture video while putting finger on the sensor surface and (b) fingerprint image acquired and saved.. 90

Figure 46 : Boundaries of the detected pupil and iris.. 95

Figure 47 : Three decision areas bounded by two thresholds, based on the overlapping area between genuine and impostor densities.. 97

Figure 48 : Verification of the face 100

Figure 49 : Verification of the fingerprint 100

Figure 50 : verification of the iris... 100

Figure 51 : final score given by the normalization and fusion of the three previous single scores 100

Figure 52 : Screenshot showing the available algorithms and options to handle them 104

Figure 53 : FAR and FRR curves plotted against threshold values for the Face modality 107

Figure 54 : Genuine and impostor matching score densities of the iris modality 108

Summary

In today's security-conscious society, biometrics-based authentication and identification have become the focus of many important applications because it is believed that biometrics can provide accurate and reliable identification.

Biometrics research and technology continue to mature rapidly, given pressing industrial and government needs and the strong support of industrial and government funding, especially in USA, China and India countries[1].

However, many of the applications warrant higher accuracy performance, which is not feasible with a single biometric today. In fact it is widely believed that fusing multiple biometrics can improve the accuracy of person identification in terms of a major reliable system that allows people to enter in certain areas. Furthermore it is thought that multibiometric approach also enable indexing of large databases and enhance coverage of the part of the population that is not able to provide any single biometric and so may not able to interface with a single biometric system.

Multiple biometrics is also naturally more robust against spoof attacks as well, because hackers have to contend with more than one biometric trait.

The aim of this thesis is to develop a prototype software in order to fuse multiple traits as evidences of an individual's identity and reach a more reliable identification. The software will be able to embed new several recognition algorithms (suitable for different traits) that answer to a specific interface. The algorithms are evaluated using some metrics and submitted to the user according to them, in order to retrieve the temporary most suitable algorithm for him.

The software will create a users database containing images and templates for each of them owning to a specific algorithm. By now it uses face, iris and fingerprint recognition algorithms, licensed by Neurotechnology (1).

The main idea is that the software will be an authentication software, and the user will be allowed to enter just giving one (or more if needed) biometric trait previously enrolled.

[1] India is currently implementing its Universal Identification (UID) program to provide a unique identification number based on biometric identifiers to each of its 1.25 billion citizens. Although it is in early stages, the UID program is already the largest biometric identification program in the world with more than 200 million people enrolled as of January 2012.

1. Introduction

Security is a word that in the last decade has been used more and more every year, also for motivations we know, and is increasingly associated to a need in social, industrial, economic and even private environments. The continuous improvement in the information technology allow us trying to better defend against threats using various modalities (video surveillance, Electronic ID-card access etc.). In the other hand, the same technological improvement allows even malicious to better attack security systems. The easiest example is the generic brute-force attack (or several its modifications using some expedientes) in order to find the right password trying all the possibilities. The today's available computing power allows to use this simple but expensive technique, in certain conditions, when some years ago was unthinkable.

Furthermore, the necessity of a reliable person identification is becoming a more and more critical element since the set of people able to get an electronic connection is enhancing rapidly.

Personal identification is the process where an individual is coupled with an identity in an univocal way, and as result they are in a two-way association.

Nowadays the most common used ways in people authentication/identification are, as we know, token-based and knowledge-based systems. In the former case, to gain access to the system, you need to own an ID-card (or a RFID chip, or a dongle) that you have to show, in the latter case you need to know an username and password and give them to the system. But both of them are quite easy to break, since an ID-card can be stolen or forgotten by an user, and usually password are not very strong because the stronger the key is, the harder it is to remember. It is quite surprising knowing that the 25% of people keep their PIN codes just in the same place (e.g. wallet) of the credit card, thus deleting all security warranties.

It is also possible to find other authentications systems:

- *Transaction authentication:*
  It looks for logical flaws when comparing known data about a user with the details of the current transaction. For example, if a user that lives in the U.S. purchases several big ticket items while logged in from an IP address determined to be from a foreign country, this is cause for concern and would require extra verification steps to ensure the purchase is not fraudulent.

- *Multifactor authentication:*
  MFA uses two or more independent sources of information to verify an identity, like:
  - Something possessed (token)
  - Something known (password)
  - Something inherent (biometric trait)

  As an easy example, ATM machine is an MFA where you need something possessed (credit card) and something known (PIN number).

- *Out-of-band authentication:*
  It uses a completely separate channel to verify an identity and allow a transaction. For example to allow a transaction made from a computer after putting a key or PIN number received at the mobile device. (2)

But in the last decade the next step towards a major security level that has received more attention has been the Biometric field. Furthermore researchers has noted that just one biometric source is no longer enough to provide a sufficient amount of reliability. The actual direction is to merge more than one biometric source, the so called Multibiometric, where actually the scientific community is focusing its own studies.

1.1. What Biometric Is

Biometrics refers to the technology for personal identification or authentication based on our physiological and/or behavioral characteristics. It literally means "measuring life" but in practical it is thought as a measure of an individual's unique physiological or behavioral characteristic and refers to the use of such known and recorded physical user’s trait to authenticate his identity, as no two individuals share the same exact physical traits.

Body type and muscle strength, for instance, can be used to distinguish gender. Moreover, the presence or absence of wrinkles around the eyes and loose facial skin suggests a person’s age. In addition, the size and shape of a person’s face, belly, thighs, and arms can determine a body habitus.

Since the beginning the human beings have unaware been recognizing each other making use of the biometric traits, first of all identifying people by watching their faces. Another example of unaware biometric recognition that we daily do is recognizing someone by listening his voice, if we cannot see him. And other sources of biometric information that we use to identify persons are for example hairs color, eyes color, height, weight or even the way someone walk (gait).
Biometrics is divided in two main categories :

- *Physiological-based* : A physiological biometric would identify by one's voice, DNA, hand print, fingerprint or facial features. In general traits in this category are stable and virtually nonalterable without severe damage to the individual.
- *Behavioral-based* : Behavioral biometrics perform the identification task by recognizing people’s behavioral patterns, such as signatures, keyboard typing, gait, and voice print. The main problem is that they all have high variations and can be difficult to measure because of influences such as stress, fatigue or illness. But in the other hand sometimes they are more acceptable to users and generally cost less to implement. (3)

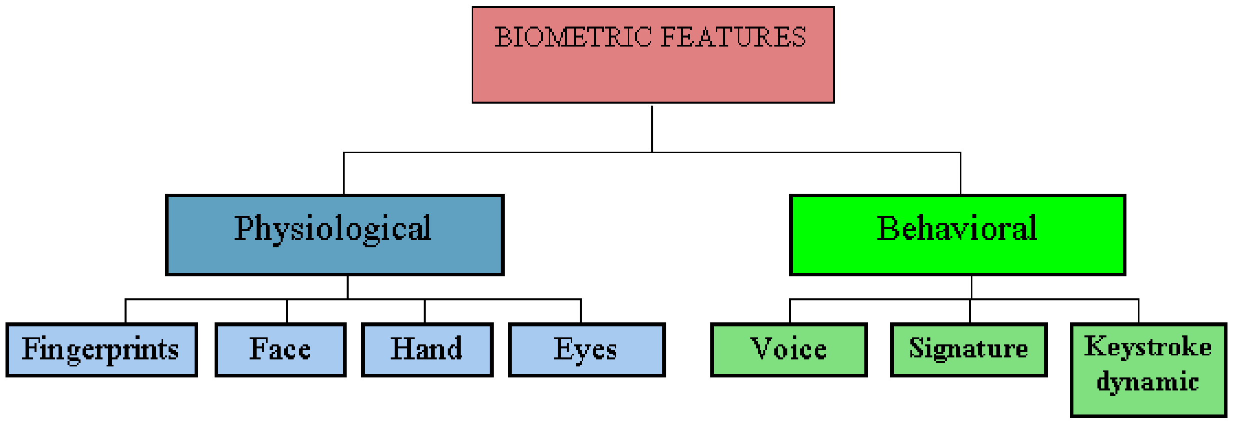


Figure 1: Examples of Physiological and Behavioral based biometrics systems

The advances in engineer and computing science, have allowed to change the way the measure of physical human traits is done, becoming targets of software development, and changing a little bit the biometrics mean in "automated methods of identifying or authenticating the identity of a living person based on a physiological or behavioral characteristic", where the key-word is in fact “automated”.

The biometrics has born as law’s needs, where the aim was to develop a reliable method to identify criminals. Alphonse Bertillon[1] has been a pioneer in using biometric traits as a method to identify malicious, but his idea it was soon obscured by a far more significant and practical discovery of the distinctiveness of the human fingerprints in the late 19th century. Later, many major law enforcement departments embraced the idea of first “booking” the fingerprints of criminals and storing it in a database. So, the leftover fingerprints at the scene of crime could be “lifted” and matched with fingerprints in the database to determine the identity of the owners. But as said before nowadays biometrics is also widely used to gain access in environments where public or private security is needed. It is moreover becoming an instrument to allow financial transactions and on-line sales.

Examples of government and civilian applications are IAFIS system (4)(Integrated Automated Fingerprint Identification System) of the FBI, the SmartGate system at the Sydney Airport based on facial recognition (5), the Privium System at Amsterdam’s Schiphol airport based on iris recognition (6) (Fig. 2), the UID program started by the UIDAI (Unique Identification Authority of India) that actually is the biggest biometrics authentication program in the world (7) and the US IDENT program (8).

[1] **Alphonse Bertillon** (April 24, 1853 – February 13, 1914) was a French police officer and biometrics researcher who created anthropometry, an identification system based on physical measurements. Anthropometry was the first scientific system used by police to identify criminals.

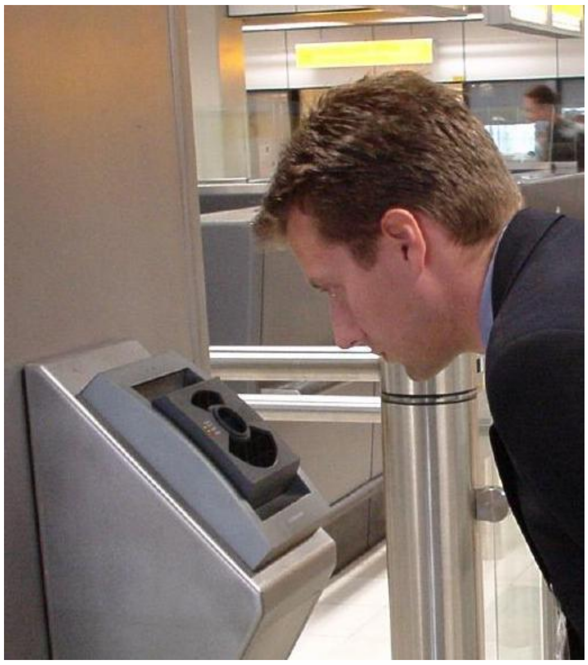


Figure 2 : An iris scan at the Schiphol Amsterdam Airport

1.2. Advantages

Biometrics overcomes the weaknesses of traditional personal identification schemes including token-based approaches and knowledge-based approaches. Follow some considerations :

- You are not asked, for example, to keep the information in a safe place to prevent theft, since you are always bringing it with you making it available whenever you need (biometric characteristics are unforgettable, you cannot lose your fingerprint or face or irises).
- Furthermore, traits cannot be easily stolen and copied[1]. (no duplication)
- Moreover, regarding to the knowledge-based approach, a password can be shared among users (allowed or not allowed). This makes the system unable to understand who the actual user is using it, in fact a credit card transaction can only validate the credit card number and the PIN, not if the transaction is conducted by the rightful owner of the credit card. A biometric reading, instead, unequivocally attaches a person to an event (most used method to solve crimes is the DNA matching).
- A biometric authentication system will eliminate the need for any documents to support your identity. Any paperwork is highly susceptible to damage, they can also be lost, but features are never lost (unless you face a traumatic accident!). (9)

[1] The difficulty in copying the physiological characteristic is not just referred to the mere operation of getting somehow the fingerprint image, for instance, of an individual, but it is mainly referred to the difficulty to use the copy with a system able to recognize if the biometric being identified is from a live person.

Thus biometrics is being more and more widely used in recently year owing to the irreproducible characteristics of the human body. The features requested to a physiological characteristic to be qualified as biometrics are :

- *Uniqueness* : means that the probability in the relevant population, that two different individuals have the same characteristic should be sufficiently low (in other words the trait features should be sufficiently different between them).
- *Universality* : means that every person should be possess that characteristic.
- *Permanence* : means that the characteristic should be reasonably invariant over the time with respect to the specific matching criterion developed for that trait.
- *Measurability* : means or concern the ease of acquisition or measurement of that characteristic.

These are not the only parameters that make a biometric trait ready to be used in a Biometric Authentication System. An organization that want to develop such a system needs to face (find the right trade-off) with other parameters that "measure" the characteristic :

- *Performance* : refers to the accuracy, robustness and speed the system can reach using that trait.
- *Acceptability* : relates to the degree of openness the users in the relevant population are willing to have their biometric trait captured and assessed. It is subjective and
- *Circumvention* : relates to the ease with which the trait could be imitated and thus bypass a biometric system which uses that characteristic.

The use of these parameters is also significant to compare different biometric technologies. Biometric consumers often finalize systems and devices to avail biometric security by comparing them on the basis of these parameters. Anyway no single biometric will meet all the requirements of every possible application.

1.3. Disadvantages

Biometrics have even some disadvantages that are not very clear and suddenly visible.

- *Integration :* Biometrics by itself is insufficient as an information security mechanism. When biometrics are a component of the internal control system, the challenge is to strategically link and integrate it with other controls to protect business systems.
- *Errors matching :* The matching process suffers from imprecise standards (such as the rigidity of thresholds to define performance precision) for the measurement of similarity. Stricter (or more relaxed) matching requirements result in higher rates of false rejections (or false acceptances). Classic approaches in fact don't suffer of false rejections since that if an individual has the right password the likelihood that he will be rejected Is zero.

- *Enrollment-dependent :* Security afforded by a biometric device is a function of controls embedded in the enrollment process. The promise of biometrics is meaningless if an enrollee is able to construct a false digital persona to access a protected system.
- *Users acceptability :* Civil libertarians stigmatize biometrics as being intrusive by nature, a potential tool for mass profiling, and a harbinger of the erosion of individual privacy. For example, certain cosmetics affect the quality and accuracy of fingerprint samples, thereby denying access to users. In such situations, users are typically asked to alter their behavioral patterns in order to facilitate the technology.
- *Legal Issues :* The lag between advances in technology and the law is especially troubling with respect to biometric technology. For instance it is still unclear whether the user or the organization that uses biometrics owns the samples and templates. The problems could be the lack of biometric standards internationally and the privacy and security issues.
- *Non Universality* : Single biometric systems could (based on a single biometric trait) suffer of the non complete universality of the related biometric feature. Some individuals could not have sufficient minutiae on their fingers needed in a fingerprint verification, or had an accident that made the trait not recognizable. This is not the case of token-based and knowledge-based approaches that are indeed suitable for everybody. (10)
- *Not still diffused* : in the case of a e-commerce company, or whatever entity that make advantages of the on-line authentication, the idea of switching to a biometric technology has a concern due to the fact that the user will asked to have specific biometric scanner devices in order to get the physical characteristic needed by the system to enroll/authenticate.

2. Biometric Systems

As mentioned above, the advances in various information processing technologies have catalized rapidly increasing interest in the development of commercial systems for biometric authentication applications.

Such systems require the users to be present at the time of authentication, deterring them from making false repudiation claims such as “I was not there at that time”. Moreover, only biometrics can provide negative identification functionality where the goal is to establish whether a certain individual is indeed enrolled in the system although the individual might deny it.

A number of anatomical and behavioral body traits can be used for biometric recognition. Examples of anatomical traits include face, fingerprint, iris, palmprint, hand geometry and ear shape. Gait, signature and keystroke dynamics are some of the behavioral characteristics that can be used for person authentication. Voice can be considered either as an anatomical or as a behavioral trait because certain characteristics of a person’s voice such as pitch, bass/tenor and nasality are due to physical factors like vocal tract shape, and other characteristics such as word or phoneme pronunciation (e.g., dialect), use of characteristic words or phrases and conversational styles are mostly learned. Ancillary characteristics such as gender, ethnicity, age, eye color, skin color, scars and tatoos also provide some information about the identity of a person. However, since these ancillary attributes do not provide sufficient evidence to precisely determine the identity, they are usually referred to as soft biometric traits. Each biometric trait has its advantages and limitations, and no single trait is expected to effectively meet all the requirements such as accuracy, practicality and cost imposed by all applications. Therefore, there is no universally best biometric trait and the choice of biometric depends on the nature and requirements of the application (11).

2.1. Components of a Biometric system

A typical biometric system is designed using the following four main components (Fig. 3):

- *A Sensor* : it is the physical device that acquires the specific trait and retrieves the data to the system. It can be sometimes coupled with a quality checker algorithm that ask for a new sample of the individual if the previous sample’s quality was not satisfactory. The biometric reader produces a digital representation (feature values) of the characteristic.
- *A feature extractor* : it gives the attention only the salient information from the acquired biometric sample to form a new representation of the biometric trait, called the *feature set*. Ideally, the feature set should be unique for each person (*extremely small inter-user similarity*) and also invariant with respect to changes in the different samples of the same biometric trait collected from the same person (*extremely small intra-class*

variability). The feature set obtained during enrollment is stored in the system database as a *template*. For example, the position and orientation of minutiae points (local ridge and valley singularities) in a fingerprint image are extracted in the feature extraction module of a fingerprint-based biometric system.

- *A Matcher module* : it uses the feature extractor to get the feature set of a query biometric sample coming from an individual, in order to compare it with a specified template. A similarity (dissimilarity) degree is then computed between them. For example, in the matching module of a fingerprint-based biometric system, the number of matching minutiae between the input and the template fingerprint images is determined and a matching score is reported.
- *A Decision module* : it uses the degree of similarity (dissimilarity) coming from the matcher module to decide on the identity of the user.

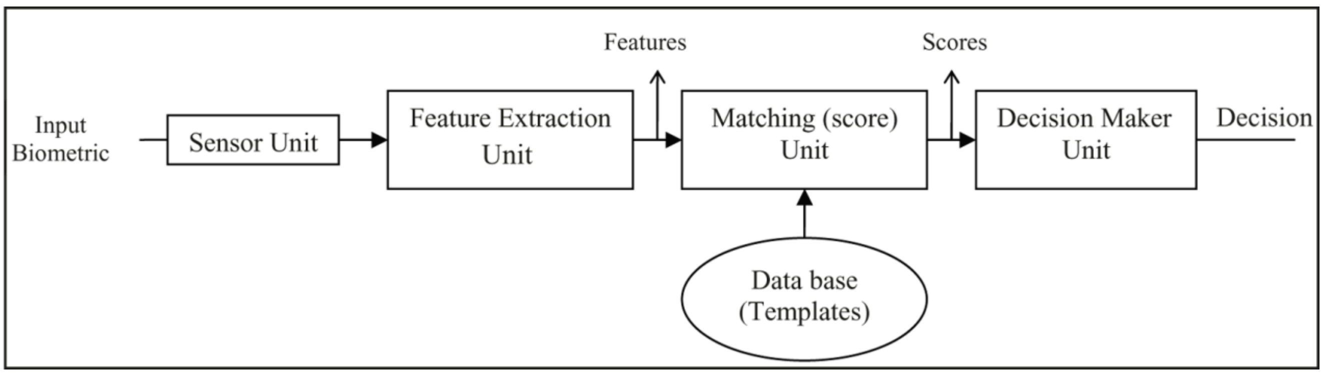


Figure 3 : Generic Biometric System components/units

Seeing it from a software point of view, the general biometric system software architecture has four major components:

- *The Graphic User Interface*: it is the interface between user and software, at which is asked to be simply, understandable and easy to use in order to cover all the type of users.
- *The underlying database* : is the database used to keep each user's information relevant for the application and the context where the system is going to work in. This component is extremely important since it contains private and sensible user's data in the form of tables and records. Here, the biometric templates of the enrolled users are stored as well
- *The Enrollment module* : It uses the feature extractor to create an individual's template (model) and store it in the database (Fig. 4). The purpose is to compare it with future query templates to give (deny) access to a genuine (impostor) user. As mentioned above, sometimes there could be a quality image checker between the Sensor and the Feature Extractor to prevent a bad enrollment phase due to a corrupted image.

A bad enrollment phase could in fact lead to a high rate of genuine user rejections for that user.

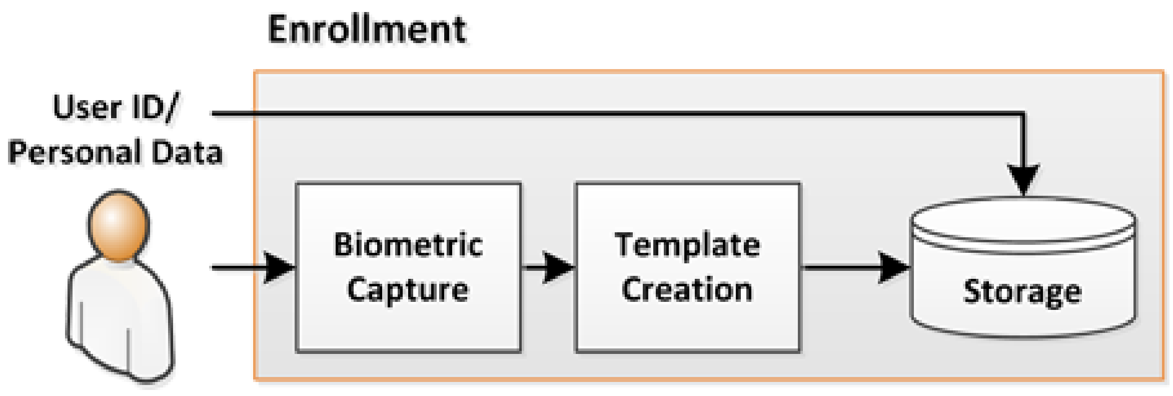


Figure 4 : Enrollment phase

- *The Authentication/Verification module* : It first uses the matching module to compare the features extracted from the query trait with the features retrieved from the database belonging to the claimed user. Then it uses the decision module to take a final decision and allow or deny the user to enter the area (or complete a financial transaction and so on).

2.2. Verification Vs Identification

A biometric system can thus be used in verification or identification mode :

- In the verification mode, the system validates a person's identity by comparing the captured biometric data with her own biometric template(s) stored in the system database. In such a mode, an individual who desires to be recognized claims an identity, usually via a PIN (Personal Identification Number), a user name, a smart card, etc., and the system conducts a one-to-one comparison to determine whether the claim is true or not (e.g., "*Does this biometric data belong to Bob?*"). Identity verification is typically used for *positive recognition,* where the aim is to prevent multiple people from using the same identity (Fig. 5). If the user's input and the template of the claimed identity have a high degree of similarity, then the claim is accepted as "genuine". Otherwise the claim is rejected and the user is considered as "impostor".
  Formally, verification can be posed as the following two-category classification problem: given a claimed identity I and a query feature set X_Q, we need to decide if $(I, \ X_Q)$ belongs to a "genuine" or "impostor" class. Let X_I be the stored template corresponding to the user I. Then usually X_I and X_Q are compared and a similarity match score *S* between them is computed. The decision rule is given by

$$(I,\ X_Q) \in \begin{cases} genuine, & if\ S \geq \eta \\ impostor, & if\ S < \eta \end{cases}$$

where η is a pre-defined threshold. Here *S* is assumed to be a similarity score, i.e., a large score indicates a good match. It is also possible for the match score to be a dissimilarity or distance score, i.e., a large score indicates a bad match between X_I and X_Q. In the latter case the inequalities in the above formula should be reversed.

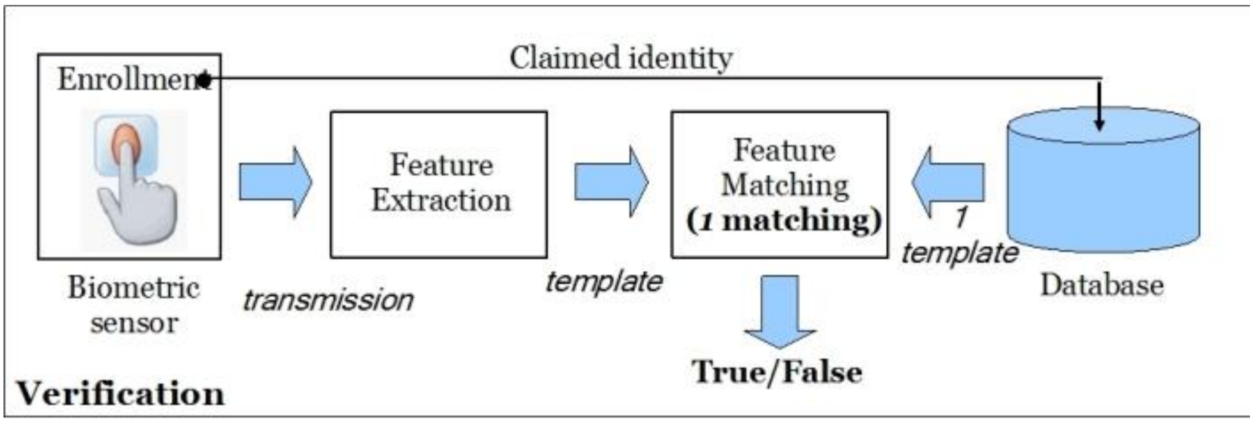


Figure 5 : Verification phase

- In the identification mode, the system recognizes an individual by searching the templates of all the users in the database for a match. Therefore, the system conducts a one-to-many comparison to establish an individual's identity (the person whose template has the highest degree of similarity with the user's input). If such a person does not exist (the subject is not enrolled in the system database) then the system will output a decision indicating that the user presenting the input is not an enrolled use. This without the subject having to claim an identity (e.g., "*Whose biometric data is this?*").
  Identification is a critical component in *negative recognition* applications where the system establishes whether the person is who he/she (implicitly or explicitly) denies to be. As an example, screening is often used at airports to verify whether a passenger's identity matches with any person on a "watch-list". The purpose of negative recognition is to prevent a single person from using multiple identities. Negative identification is critical in applications such as welfare disbursement to prevent a person from claiming multiple benefits under different names.
  Identification may also be used in positive recognition for convenience (the user is not required to claim an identity). While traditional methods of personal recognition such as passwords, PINs, keys, and tokens may work for positive recognition, negative recognition can only be established through biometrics (Fig. 6).
  As for verification problem, the identification problem can be formally stated as follows: given the feature set X_Q, we need to find the identity I of the user, where $I \in \{I_1, I_2, \ldots.,$

$I_N, I_{N+1}\}$. Here $I_1, I_2, \ldots.,\ I_N$ correspond to the identities of the *N* users enrolled in the database, and I_{N+1} indicates the case where no suitable identity can be determinate for the given query.
If $X_{I\ n}$ is the stored template corresponding to the user I_N, and S_n is the similarity match score computed after comparing X_Q and $X_{I\ n}$ for *n* = 1, 2, …., *N,* the decision rule for identification is :

$$X_Q \in \begin{cases} I_{n_0} & if\ n_0 = \arg\max_n S_n \quad and\ \ S_{n_0} \geq \eta \\ I_{N+1} & otherwise \end{cases}$$

where η is a pre-defined threshold. The identification problem can become a significant more challenge task rather than the verification, since the number of enrolled users in template databases can be quite large. The UID Indian program is potentially going to enroll and store one billion and two hundred millions of users.

Accennare al fatto che esistono tecniche per velocizzare l'identificazione scegliendo sottoinsiemi mirati dell'insieme degli utenti registrati, perché possiedono caratteristiche comuni al query template.

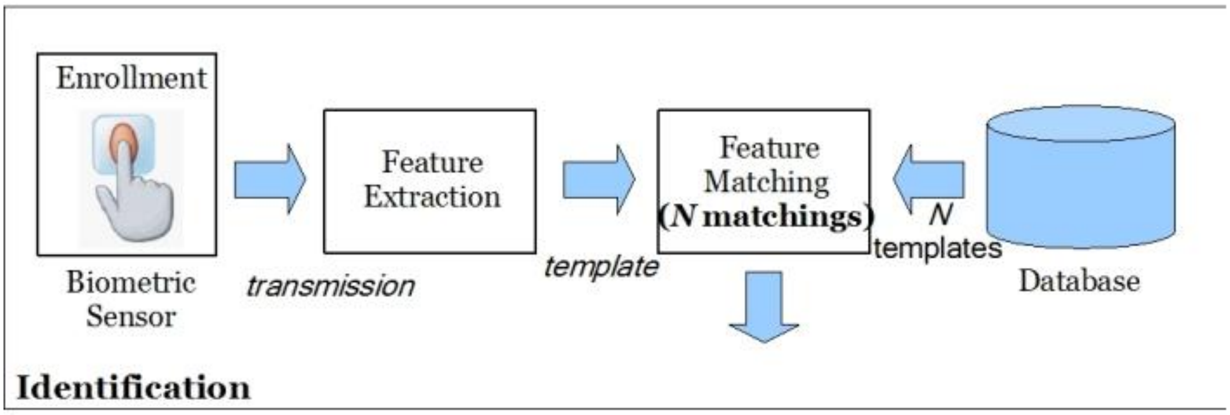


Figure 6 : Identification phase

Above we talked about two different type of recognition: positive and negative:

The positive recognition is simple the checking process to confirm if the individual can gain access to the system, since he shows some data that for the system are enough to let him pass.
The negative recognition is the process through which the system can say if the individual is who he is not saying to be. In other words, if the individual is Bob, the biometric system can state that the user can't be anyone else except Bob.

A knowledge-based or token-based system cannot perform a negative recognition because the fact that the individual is using or presenting some data, is not enough to claim that "He is Bob".

Biometric trait instead allow the system to state “since you have this data, you are Bob, and you can’t be anyone different from Bob”, due to the difficulty of copying and sharing the physiological characteristic.

2.3. Measures of Biometrics performance

Evaluating an identification biometric system in an objective way is still an open problem today. There is not a standard method to follow and which can state if a biometric application is good or not under all the aspects. In fact every such a system has pros and cons that make the developer to find a trade off of them.

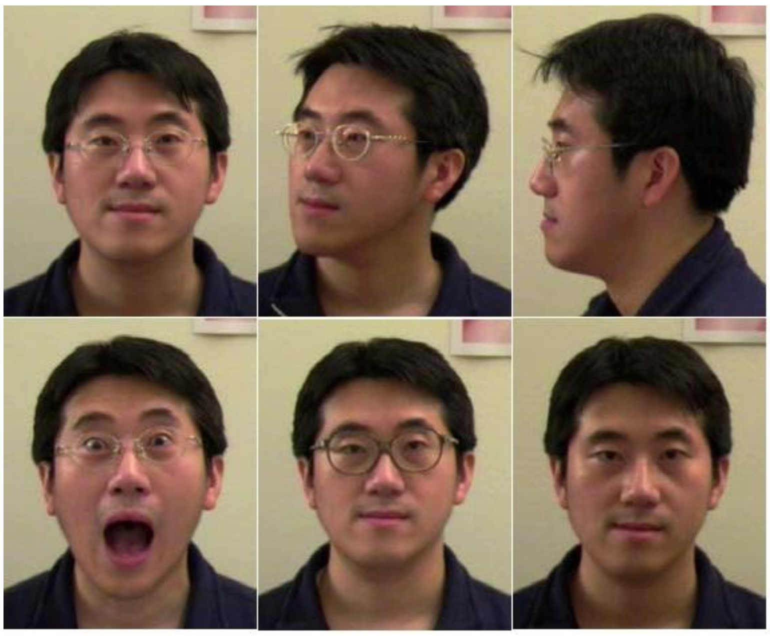
At first we have to consider if the system is going to operate most in verification or identification mode. It is more difficult to design an identification system than to design a verification system. For the latter one the most important issue is to meet a high level of accuracy, since only one-to-one comparison is made and the response time requirement is not a challenge. In the former one both the accuracy and the speed are critical because it has to scan all the database templates (worst case) to find the user identity. Moreover the speed intrinsically depends on the physical characteristic that is used in the system. For example a real time face recognition is more feasible than a real time fingerprint recognition, due to (*i*) face comparison is a relatively less expensive operation and (*ii*) efficient indexing techniques are available and the performance is admissible.

2.3.1. Errors of a Biometric system

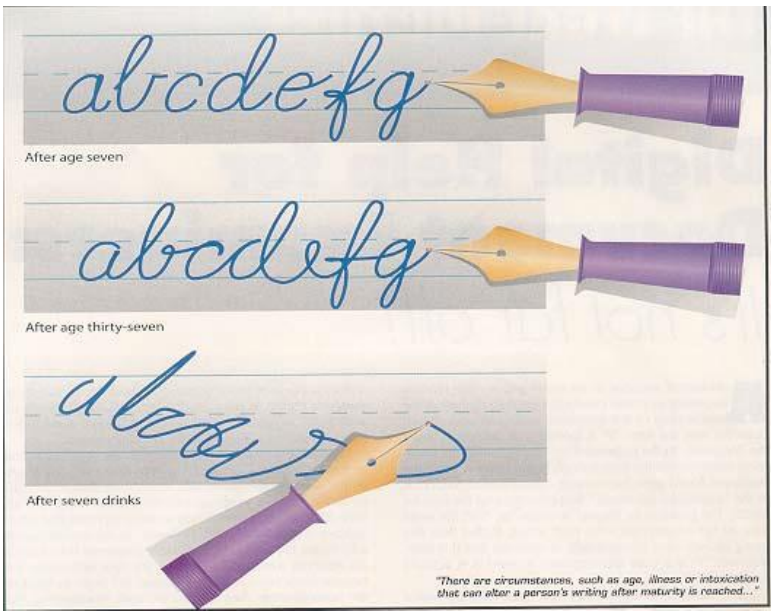
Due to *intra-class variations* in biometric characteristics, the identity can be established only with certain confidence. Intra-class variations is the variability observed in the biometric feature set of an individual’s trait (Fig.7). Samples of the same biometric trait of a user obtained over a period of time can differ dramatically (11), and that’s the reason why the template created during the enrollment phase could be significantly different from template proposed in verification phase. For this reason a large intra-class variation can lead to reject a registered user with high probability.
Follow some examples that can lead to large intra-class variations :

- Fingers, Palm
  - Placement/pose variations of the finger/palm on the sensor
  - Applied finger, palm pressure
  - Skin condition
- Face
  - Head motion caused by unsupervised user
  - Facial expressions
  - Cosmetics
  - Hair styles

- Commons (Ears, Iris etc.)
  - Feature extractor errors
  - The use of poor camera quality in face, hand, iris and ear recognition
  - Illumination variation



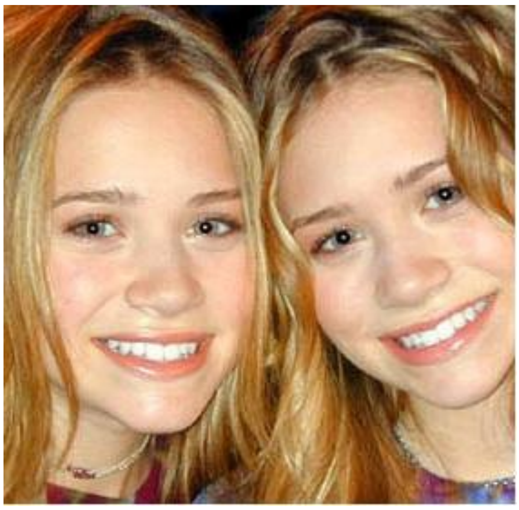
(a)


abcdefg
After age seven
abcdefg
After age thirty-seven
After seven drinks
"There are circumstances, such as age, illness or intoxication that can alter a person's writing after maturity is reached..."

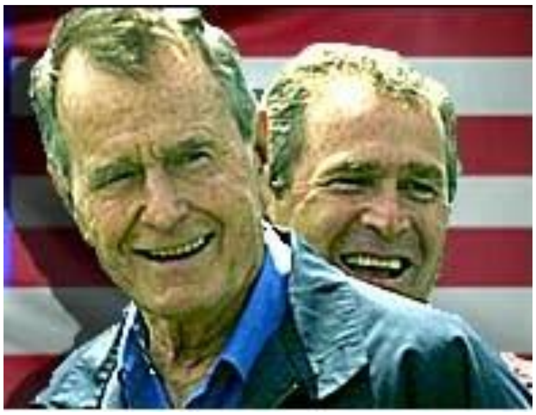
(b)

Figure 7 : Examples of intra-class variations in (a) face recognition (b) signature recognition

Biometrics systems has also to face with *inter-class similarities*, that is the similarities between different users referred to the same biometric trait, or more formally, the overlap of features space corresponding to multiple individuals (12). For example, some pairs of individuals can have nearly identical facial appearance due to genetic factors (e.g., father and son, identical twins, etc.) (Fig. 8). Then a large inter-class similarities can lead to accept an impostor not registered in the database with high probability.



(a)



(b)

Figure 8 : Example of inter-class similarity of (a) twins and (b) father and son

The confidence associated with different decisions may be characterized by the genuine distribution and the impostor distribution scores. They represent the density functions of all the genuine and impostor scores. A score is labeled as genuine if it is computed by the system after comparing two templates coming from the same user (mate samples). A score is instead labeled as impostor if it is computed by the system after comparing two templates coming from two different users (non mate samples) (Fig. 9).

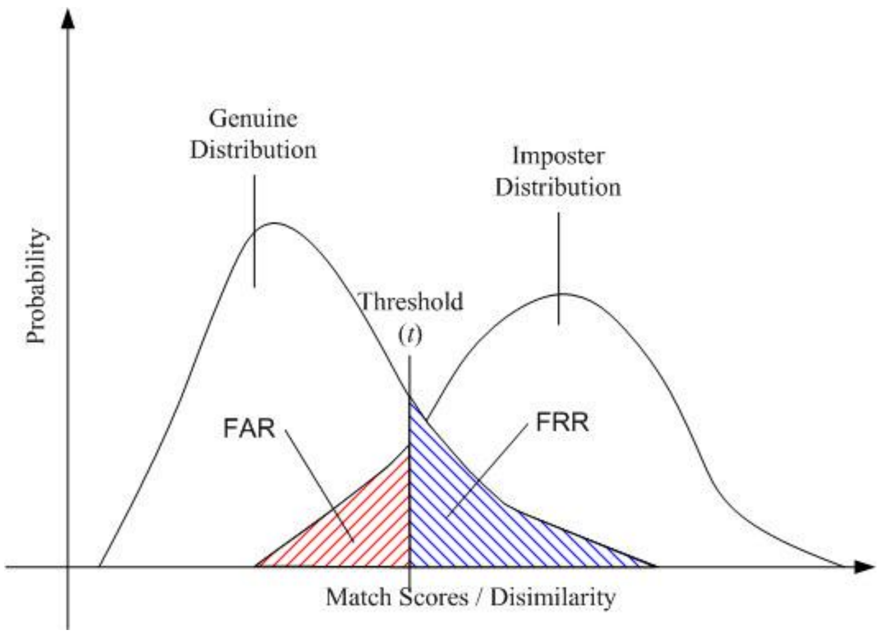


Figure 9 : An example of Genuine and Impostor distribution scores.
In the X axis are posed the scores we can obtain from the system, referring to distance/dissimilarity scores. In the Y axis is shown the probability.
The region under the Genuine Distribution curve, refereed as FAR, represents the region where an impostor's score can be recognized as coming from a genuine user.
The region under the Impostor Distribution curve, referred as FRR, represents the region where a genuine's score can be recognized as coming from an impostor user

2.3.2. Performance metrics

The classification problem is a mapping of instances into a certain class/group where the classifier boundary between classes must be determined by a threshold value. For instance, to determine whether a person has hypertension based on blood pressure measure.
The most simple classification problem is the two-class prediction problem, also known as binary classification, in which the outputs are labeled either as *positive* or *negative* class. At the same time the input instances can be either *positive* or *negative* as well, so leading to four possible cases (Fig. 10) :

- TP : If the output is *positive* and the input instance is also *positive*, then the outcome is said to be a True Positive.
- TN : If the output is *negative* and the input instance is also *negative*, then the outcome is said to be a True Negative.
- FP : If the output is *positive* while the input instance is *negative*, then the outcome is said to be a False Positive.
- FN : If the output is *negative* while the input instance is *positive*, then the outcome is said to be a False Negative.

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Figure 10 : Binary classification confusion matrix

Consequently several performance evaluation metrics are available:

- Sensitivity : The probability with which the system correctly identify a *positive* input instance, that is, the number of True Positive outcomes on the total *positive* attempts.
- Accuracy : The probability with which the input instance (whether *positive* or *negative*) is correctly identified, that is, the number of True Positive plus True Negative outcomes on the total number of attempts.
- Specificity : 1-FP/N The probability with which the system correctly identify a *negative* input instance, that is, the number of True Negative outcomes on the total *negative* attempts.
- False Positive Rate : The probability with which the system falsely identify a *negative* input instance, that is, the number of False Positive outcomes on the total *negative* attempts.

In the Biometric recognition field, the biometric authentication is a binary classification problem where an user will be labeled as "genuine" or "impostor" individual.
Both of them can be a right or wrong choice Then there are a total of four possible outcomes: (*i*) a real genuine user that is accepted as "genuine individual", (*ii*) a real genuine user that is labeled as an "impostor individual", (*iii*) an impostor user that is labeled as a "genuine individual" and *(iv*) an impostor that is recognized as an "impostor individual".
Outcomes (*i*) and (*iv*) are correct while (*ii*) and (*iii*) are incorrect.

The (*ii*) case is known as FRR (False Rejecton Rate), or as FNMR (False Non-Match Rate), that is the probability that a genuine user being rejected as an impostor. When the intra-class variation is large, two samples of the same biometric trait of an individual (mate samples) may not be recognized as a match, leading to a false reject error . A FRR of 1%, for instance, states that on average 1 in 100 genuine attempts do not succeed. A majority of the false reject errors are usually due to incorrect interaction of the user with the biometric sensor and can be easily rectified by allowing the user to present his/her biometric trait again, as in a password-based authentication system when an user makes a mistake while entering a password and he is allowed to reenter it.

On the other hand, the (*iii*) case is called FAR (False Acceptance Rate), or FMR (False Match Rate), that is an impostor being recognized as a genuine individual. A false match occurs when two samples from different individuals (non-mate samples) are incorrectly recognized as a match, maybe due to large inter-class similarity. For an impostor, guessing the mean to circumvent the system is harder than a password-based system for instance. This due to the fact that firstly he need a features space and domain knowledge to get the correct input values (in a password-based system he just need to know how to count), secondly he need to attack one of the physical components in the biometric system, that can be made very difficult by appropriate techniques such as liveness detection, cryptographic protocols and secure code execution.

A verification system makes a decision by comparing the match score S to a threshold η. Therefore, FRR can be defined as the proportion of genuine scores that are less (greater in the case of dissimilarity scores) than the threshold η and FAR can be defined as the fraction of impostor scores that are greater (less in the case of dissimilarity scores) than or equal to η (Fig. 9).

Let $f_{gen}(s) = p(S = s|genuine)$ and $f_{imp}(s) = p(S = s|impostor)$ be the probability density functions of the genuine and impostor scores respectively. The FAR and FRR of the biometric system are given by :

$$FAR(\eta) = p(S \geq \eta|impostor) = \int_{\eta}^{\infty} f_{imp}(s)ds$$

$$FRR(\eta) = p(S < \eta|genuine) = \int_{-\infty}^{\eta} f_{gen}(s)ds$$

Both FRR and FAR are functions of the system threshold η. If the threshold is increased, FAR will decrease but the FRR will increase and vice versa. Hence, for a given biometric system, it is not possible to decrease both these errors simultaneously by varying the threshold (Fig. 11).

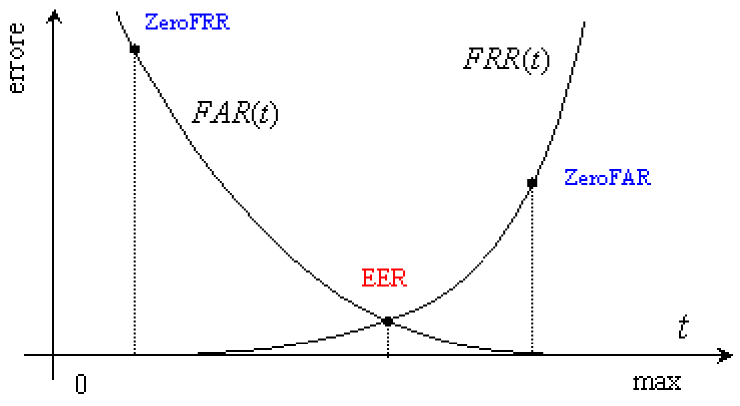


Figure 11 : FAR and FRR percentages to vary the threshold

The FAR and FRR of a biometric system at different values of threshold η, can be summarized in the form of Receiver Operating Characteristic (ROC) curve.
A general ROC curve is defined as a graph where the *x* and *y* axis are the FPR (1-specificity) and TPR (sensitivity) respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). This graph is useful to understand if the binary classifier under test is behaving well or not.
In Figure 12 is shown such a defined ROC curve. The best possible prediction method would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 100% sensitivity (no false negatives) and 100% specificity (no false positives, that is 0% FPR). The (0,1) point is also called a *perfect classification*.
A completely random guess would instead give a point along a diagonal line (the so-called *line of no-discrimination*) from the left bottom to the top right corners (regardless of the positive and negative base rates). An intuitive example of random guessing is a decision by flipping coins (heads or tails). As the size of the sample increases, a random classifier's ROC point migrates towards (0.5,0.5) point.
The diagonal divides the ROC space. Points above the diagonal represent good classification results (better than random), points below the line poor results (worse than random).
Note that the output of a consistently poor predictor could simply be inverted to obtain a good predictor (13).

In Figure 12 four prediction results are plotted from 100 positive and 100 negative instances

A

TP= 63	FP= 28	91
FN= 37	TN= 72	109
100	100	200

TPR = 0.63
FPR = 0.28
ACC = 0.68

B

TP= 77	FP= 77	154
FN= 23	TN= 23	46
100	100	200

TPR = 0.77
FPR = 0.77
ACC = 0.50

C

TP= 24	FP= 88	112
FN= 76	TN= 12	88
100	100	200

TPR = 0.24
FPR = 0.88
ACC = 0.18

C'

TP= 76	FP= 12	88
FN= 24	TN= 88	112
100	100	200

TPR = 0.76
FPR = 0.12
ACC = 0.82

The result of method A clearly shows the best predictive power among A, B, and C. The result of B lies on the random guess line (the diagonal line), and it can be seen in the table that the accuracy of B is 50%. However, when C is mirrored across the center point (0.5,0.5), the resulting method C′ is even better than A. This mirrored method simply reverses the predictions of whatever method or test produced the C contingency table. Although the original C method has negative predictive power, simply reversing its decisions leads to a new predictive method C′ which has positive predictive power. When the C method predicts p or n, the C′ method would predict n or p, respectively. In this manner, the C′ test would perform the best. The closer a result from a contingency table is to the upper left corner, the better it predicts, but the distance from the random guess line in either direction is the best indicator of how much predictive power a method has. If the result is below the line (i.e. the method is worse than a random guess), all of the method's predictions must be reversed in order to utilize its power, thereby moving the result above the random guess line.

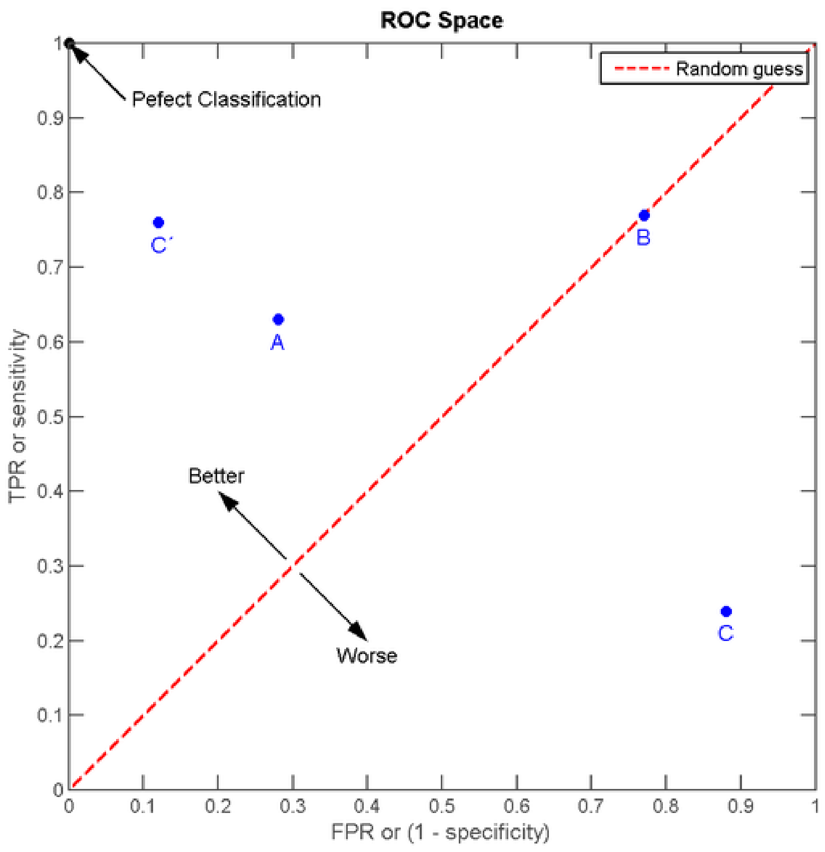
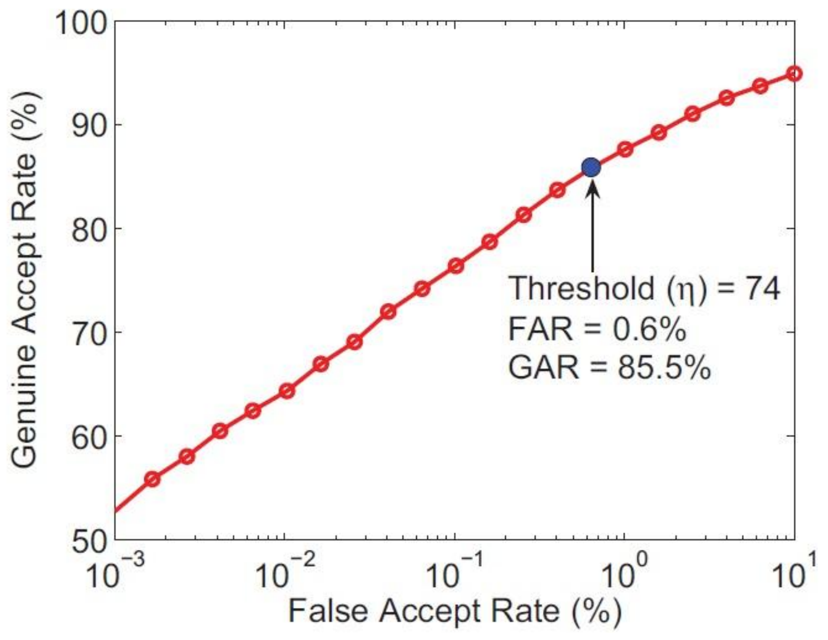


Figure 12 : ROC Space. The diagonal line represents a random binary classifier where for every decision it has 50% of probability to be correct or incorrect.

In the field of biometric systems, the ROC curve will be plotted using FAR and GAR (Genuine Acceptance Rate = 1-FRR) as *x* and *y* axis respectively. Every point in this graph (FAR;1-FRR) depends on a threshold value (Fig. 12).
Equal Error Rate (EER) is the point in a ROC curve where the FAR equals the FRR. The EER is a quick way to compare the accuracy of devices with different ROC curves. A lower EER value generally indicates better performance. But EER reflects the accuracy of a system only at a single point. If the system is expected to operate at that single point, where FAR and FRR are equal, then the system with smaller EER should be chosen to be deployed. Nevertheless, the ability to reject imposters is considered to be more important than the ability to match authorized users in many applications. That is why; many biometric systems are not operated at that single point. Instead they are usually operated at a point where FAR is smaller than FRR. Therefore, different biometric systems should be compared at the desired operating point in order to decide correctly while selecting the one to be deployed. To illustrate, if it is desired to have a biometric system operating at a FAR of 0.1 %, then the biometric system with the smallest FRR when FMR is 0.1 % should be preferred. In brief,

EER should be considered to give an idea about the system accuracy but it should not be only guideline while deciding the biometric system to be deployed. (14)



(b)

Figure 13 : Receiver operating characteristic (ROC) curve for a matcher which plots the GAR against FAR on a semi-logarithmic scale.

We also need to talk about the FTER (Failure To Enroll Rate), that is defined as the probability that an individual will be unable to enroll in a biometric system. There are many causes, related to both the environment and individuals, resulting in failure-to-enroll. Poor lighting in face recognition applications and background noise in voice recognition applications are examples for the environmental causes of FTER. There are also causes of FTER which are related to individuals. To illustrate, people working in construction tend to wear down their fingerprints. Beside the occupation, age and ethnicity of individuals can also be considered as causes of FTER related to individuals. FTER can be drastically reduced by informing enrollees in order to get best biometric templates. For example, informing individuals about the proper placement of their fingers in a fingerprint recognition application or asking them to lengthen a pass phrase in a voice recognition application will be useful in reducing FTER. In general, ensuring proper placement and interaction with biometric acquisition devices significantly decrease FTER without requiring any changes in the core biometric processes. (14)

2.4. Biometric Traits

A number of physiological and behavioral features have been proposed and proved to be valid biometric characteristics. But every trait has its limitations and weaknesses, as well as its strengths, and no one single biometric is expected to completely distinguish every individual[1]. The choice depends on the application and its operational mode as well as the biometric characteristic. In other words, no biometric is optimal, no single technique can out-perform all the others in all operational environments.

2.4.1. DNA

Deoxyribo Nucleic Acid (DNA) is the one-dimensional ultimate unique code for one's individuality except for the fact that identical twins have identical DNA patterns. It is, however, currently used mostly in the context of forensic applications for person recognition. Three issues limit the utility of this biometrics for other applications: (i) contamination and sensitivity: it is easy to steal a piece of DNA from an unsuspecting subject that can be subsequently abused for an ulterior purpose; (ii) automatic real-time recognition issues: the present technology for DNA matching requires cumbersome chemical methods (wet processes) involving an expert's skills and is not geared for on-line non-invasive recognition; (iii) privacy issues: information about susceptibilities of a person to certain diseases could be gained from the DNA pattern and there is a concern that the unintended abuse of genetic code information may result in discrimination, e.g., in hiring practices.

2.4.2. Ear

It has been suggested that the shape of the ear and the structure of the cartilegenous tissue of the pinna are distinctive. The ear recognition approaches are based on matching the distance of salient points on the pinna from a landmark location on the ear. The features of an ear are not expected to be very distinctive in establishing the identity of an individual.

2.4.3. Face

Face recognition is a non-intrusive method, and facial images are probably the most common biometric characteristic used by humans to make a personal recognition. The applications of facial recognition range from a static, controlled "mug-shot" verification to a dynamic, uncontrolled face identification in a cluttered background (e.g., airport). The most popular approaches to face recognition are based on either *(i)* the location and shape of facial attributes, such as the eyes, eyebrows, nose, lips, and chin and their spatial relationships, or *(ii)* the overall (global) analysis of the face image that represents a face as a weighted combination of a number of canonical faces. While the verification performance of the face recognition systems that are commercially available is reasonable [34], they impose a number of restrictions on how the facial images are obtained,

[1] The real mean of referring to a biometric characteristic as a way to univocally identify an individual is that it is statistically very difficult that two users have the same template as result of the feature extraction phase, but not impossible.

sometimes requiring a fixed and simple background or special illumination. These systems also have difficulty in recognizing a face from images captured from two drastically different views and under different illumination conditions. It is questionable whether the face itself, without any contextual information, is a sufficient basis for recognizing a person from a large number of identities with an extremely high level of confidence [29]. In order that a facial recognition system works well in practice, it should automatically *(i)* detect whether a face is present in the acquired image; *(ii)* locate the face if there is one; and *(iii)* recognize the face from a general viewpoint (i.e., from any pose).

2.4.4. Facial, hand and hand vein infrared thermogram

The pattern of heat radiated by human body is a characteristic of an individual and can be captured by an infrared camera in an unobtrusive way much like a regular (visible spectrum) photograph. The technology could be used for covert recognition. A thermogram-based system does not require contact and is non-invasive, but image acquisition is challenging in uncontrolled environments, where heat emanating surfaces (e.g., room heaters and vehicle exhaust pipes) are present in the vicinity of the body. A related technology using near infrared imaging is used to scan the back of a clenched fist to determine hand vein structure. Infrared sensors are prohibitively expensive which is a factor inhibiting wide spread use of the thermograms.

2.4.5. Fingerprint

Humans have used fingerprints for personal identification for many centuries and the matching accuracy using fingerprints has been shown to be very high. A fingerprint is the pattern of ridges and valleys on the surface of a fingertip, the formation of which is determined during the first seven months of fetal development. Fingerprints of identical twins are different and so are the prints on each finger of the same person. Today, a fingerprint scanner costs about US $20 when ordered in large quantities and the marginal cost of embedding a fingerprint-based biometric in a system (e.g., laptop computer) has become affordable in a large number of applications. The accuracy of the currently available fingerprint recognition systems is adequate for verification systems and small- to medium-scale identification systems involving a few hundred users. Multiple fingerprints of a person provide additional information to allow for large-scale recognition involving millions of identities. One problem with the current fingerprint recognition systems is that they require a large amount of computational resources, especially when operating in the identification mode. Finally, fingerprints of a small fraction of the population may be unsuitable for automatic identification because of genetic factors, aging, environmental, or occupational reasons (e.g., manual workers may have a large number of cuts and bruises on their fingerprints that keep changing).

2.4.6. Hand and finger geometry

Hand geometry recognition systems are based on a number of measurements taken from the human hand, including its shape, size of palm, and lengths and widths of the fingers. Commercial hand geometry-based verification systems have been installed in hundreds of locations around the world. The technique is very simple, relatively easy to use, and inexpensive. Environmental factors such as dry weather or individual anomalies such as dry skin do not appear to have any negative effects on the verification accuracy of hand geometry-based systems. The geometry of the hand is not known to be very distinctive and hand geometry-based recognition systems cannot be scaled up for systems requiring identification of an individual from a large population. Further, hand geometry information may not be invariant during the growth period of children. In addition, an individual's jewelry (e.g., rings) or limitations in dexterity (e.g., from arthritis), may pose further challenges in extracting the correct hand geometry information. The physical size of a hand geometry-based system is large, and it cannot be embedded in certain devices like laptops. There are verification systems available that are based on measurements of only a few fingers (typically, index and middle) instead of the entire hand. These devices are smaller than those used for hand geometry, but still much larger than those used in some other biometrics (e.g., fingerprint, face, voice).

2.4.7. Iris

The iris is the annular region of the eye bounded by the pupil and the sclera (white of the eye) on either side. The visual texture of the iris is formed during fetal development and stabilizes during the first two years of life. The complex iris texture carries very distinctive information useful for personal recognition. The accuracy and speed of currently deployed iris-based recognition systems is promising and point to the feasibility of large-scale identification systems based on iris information. Each iris is distinctive and, like fingerprints, even the irises of identical twins are different. It is extremely difficult to surgically tamper the texture of the iris. Further, it is rather easy to detect artificial irises (e.g., designer contact lenses). Although, the early iris-based recognition systems required considerable user participation and were expensive, the newer systems have become more user-friendly and cost-effective.

2.4.8. Keystroke

It is hypothesized that each person types on a keyboard in a characteristic way. This behavioral biometric is not expected to be unique to each individual but it offers sufficient discriminatory information to permit identity verification. Keystroke dynamics is a behavioral biometric; for some individuals, one may expect to observe large variations in typical typing patterns. Further, the keystrokes of a person using a system could be monitored unobtrusively as that person is keying in information.

2.4.9. Retinal scan

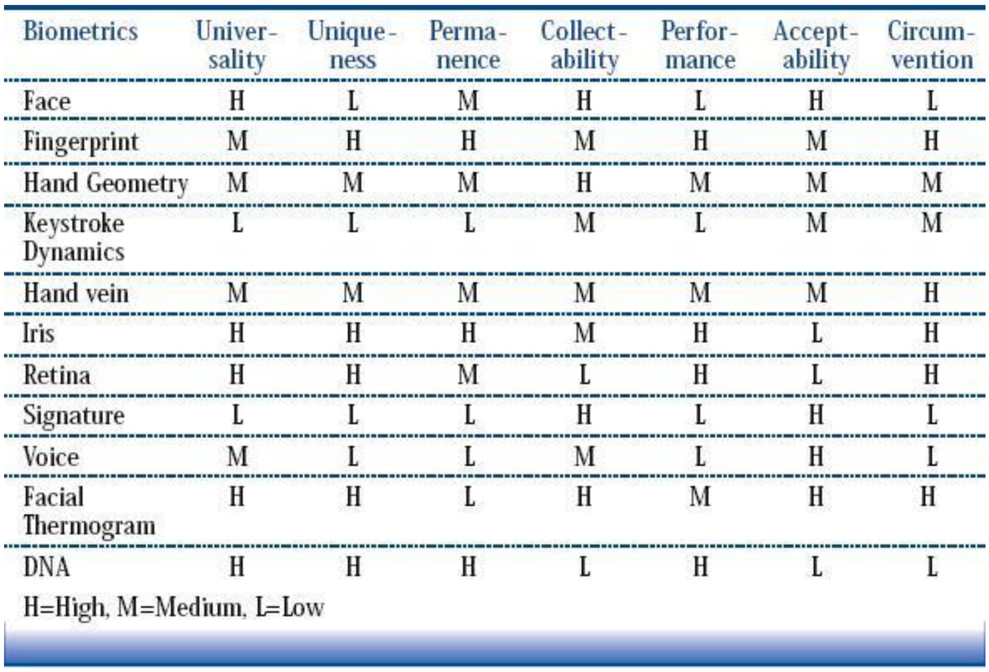
The retinal vasculature is rich in structure and is supposed to be a characteristic of each individual and each eye. It is claimed to be the most secure biometric since it is not easy to change or replicate the retinal vasculature. The image acquisition requires a person to peep into an eye-piece and focus on a specific spot in the visual field so that a predetermined part of the retinal vasculature could be imaged. The image acquisition involves cooperation of the subject, entails contact with the eyepiece, and requires a conscious effort on the part of the user. All these factors adversely affect the public acceptability of retinal biometric. Retinal vasculature can reveal some medical conditions, e.g., hypertension, which is another factor deterring the public acceptance of retinal scan based biometrics.

2.4.10. Signature

The way a person signs her name is known to be a characteristic of that individual. Although signatures require contact with the writing instrument and an effort on the part of the user, they have been accepted in government, legal, and commercial transactions as a method of verification. Signatures are a behavioral biometric that change over a period of time and are influenced by physical and emotional conditions of the signatories. Signatures of some people vary substantially: even successive impressions of their signature are significantly different. Further, professional forgers may be able to reproduce signatures that fool the system.

2.4.11. Voice

Voice is a combination of physiological and behavioral biometrics. The features of an individual’s voice are based on the shape and size of the appendages (e.g., vocal tracts, mouth, nasal cavities, and lips) that are used in the synthesis of the sound. These physiological characteristics of human speech are invariant for an individual, but the behavioral part of the speech of a person changes over time due to age, medical conditions (such as common cold), emotional state, etc. Voice is also not very distinctive and may not be appropriate for large-scale identification. A text-dependent voice recognition system is based on the utterance of a fixed predetermined phrase. A text-independent voice recognition system recognizes the speaker independent of what she speaks. A text-independent system is more difficult to design than a text-dependent system but offers more protection against fraud. A disadvantage of voice-based recognition is that speech features are sensitive to a number of factors such as background noise. Speaker recognition is most appropriate in phone-based applications but the voice signal over phone is typically degraded in quality by the microphone and the communication channel.



Biometrics	Univer-sality	Unique-ness	Perma-nence	Collect-ability	Perfor-mance	Accept-ability	Circum-vention
Face	H	L	M	H	L	H	L
Fingerprint	M	H	H	M	H	M	H
Hand Geometry	M	M	M	H	M	M	M
Keystroke Dynamics	L	L	L	M	L	M	M
Hand vein	M	M	M	M	M	M	H
Iris	H	H	H	M	H	L	H
Retina	H	H	M	L	H	L	H
Signature	L	L	L	H	L	H	L
Voice	M	L	L	M	L	H	L
Facial Thermogram	H	H	L	H	M	H	H
DNA	H	H	H	L	H	L	L

H=High, M=Medium, L=Low

Figure 14 : Comparison among several biometric modalities

2.5. Choosing a Biometric trait

The choice of a particular modality depends on the application for which it is being to use, that in turn depends on some factors such as

- *Authentication/Verification mode*: If the biometric is used in an identification role, it should have fast computability and be more unique in general. Moreover achieving the same accuracy in an identification system as in a verification system is a much harder problem due to the large number of comparisons that are required to be performed. Consider that airport authorities are looking for FBI's 100 most wanted criminals (database size of 100) and the state-of-the-art fingerprint verification system operates at 1% FRR and 0.001% FAR, i.e., if this system was deployed as a verification system, the system would fail to match the correct users 1% of the time and erroneously verify wrong users 0.001% of the time. Let us consider the outcome of the same system when deployed as an identification system. While the identification FRR will still be 1%, the identification FAR will be ~ 100×0.001%=0.1%. This means that while the system has a 99% chance of catching a criminal, it will produce large number of false alarms (e.g., assuming that 200,000 people may use a major US airport in a day, the system will produce 200 false alarms!). (15)
- *Ease of use* : Some users could have problems in understanding how to interface with the system, in particular with the sensor which has to acquire the trait. For example an individual could not understand how far and how to position the head from the camera in a face/iris/ear recognition system. The capturing device should be intuitive to use.

- *Incidence of errors due to environment* : Some environment conditions such as variation in light intensity in face recognition, high level of external noise in voice recognition and absence of the biometric trait required are factors that can spoil the enrollment phase. It is needed to take into account how much these factors can influence the system, it they are temporary or not, and how fast the trait is going to change over time.
- *Accuracy* : How much stringent the security requirements are. A biometric system seldom encounters a sample of user's biometric trait that is exactly the same as the template enrolled for him. This results in a number of errors as discussed in section 2.3.2 and thereby limits the system accuracy. It has to be clear if the system need to pay more attention denying access to impostors (low FAR) or suddenly allow genuine to access (low FRR) and find a trade-off between them since the two errors are correlated.
- *Cost* : This is a worldwide factor that afflicts almost any decision in a business environment. There are several components included in the cost component :
  - The capturing device
  - Work load of back-end processing for the database maintenance
  - Research and test on the biometric system (choosen modality)
  - Costs of system installation
  - Costs of the integration with the user system
  - Costs of user trainings
  - Costs of exceptions: users that can't be enrolled due to physiological reasons (i.e. lacks of fingerprints
  - Costs of system maintenance
- *User acceptability* : This is a metric which depends on the cultural and social norms of the place at which the biometric is going to be used. People should be comfortable using a biometric, the less intrusive, the more accepted it is. This is especially important to consider when using a biometric for commercial applications. Also, people are generally skeptical to give the more secure biometric modalities like fingerprints, iris and retina patterns at their local supermarkets for identification.
- *Duration of the application use* : If the application is going to be used for a long time, it would be necessary that the biometric choice would be relatively more stable with time.
- *Stability of the biometric technology* : Businesses should consider the stability of a biometric technology as well, taking into account the maturity and standardization of a technology. Moreover they should pay attention to the government support (i.e. laws about privacy), the vendor reliability and the market share. Well established technologies has an higher level of stability. (16)

2.6. Limitations of biometrics

Biometrics is gaining more and more attention widely in the world, and a large number of its real-world applications have been successfully developed. However there are several unsolved or half-solved problems.
Limitations of unimodal biometric systems can be separated into three major categories: *accuracy*, *scalability* and *usability*.

2.6.1. Accuracy

A biometric system seldom encounters a sample of a user's biometric trait that is exactly the same it used to enroll him. This result in a number of errors as discussed in sections 2.3.1 and 2.3.2 that affects the system accuracy (17).

- *Noisy sensor data* : the recognition accuracy is highly sensitive to the quality of biometric input and noisy data can result in a significant reduction of it. For example accumulation of dirt on a fingerprint sensor can compromise the quality of the fingerprint image, as well as failing to properly focus a camera can lead to blurred iris and face images as shown in Figure 15.
- *Non-universality* : Not all biometric traits are truly universal. In fact not all the users can give the biometric trait requested by the system, due to accidents, physiological problems or years of manual works. The non-universality leads to a major FTER
- *Inter-class similarity* : As mentioned above, the overlap of biometric samples coming from different users in the feature space, increases the FAR, since two individuals can be recognized as the same one. This problem is also known as *lack of uniqueness*.
- *Lack of invariant representation* : Variations in samples acquired during different sessions may be due to improper interaction of the user with the sensor (e.g. rotation, translation and applied finger pressure, poses and facial expressions), changes in the ambient environmental conditions (illumination) , use of different sensors in enroll and verification phase and inherent changes of the biometric trait due to aging. Large intra-class variations result to a low GAR (high FRR).

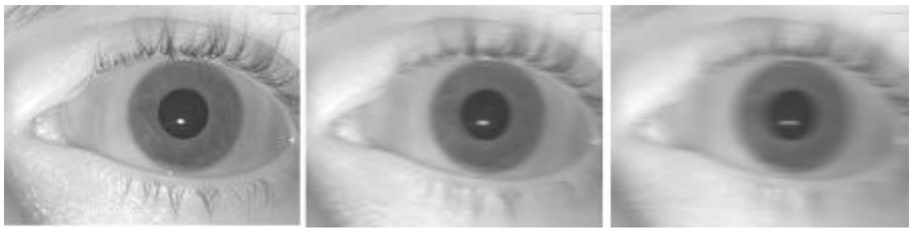


Figure 15 : Three iris images more and more blurred from left to right

2.6.2. Scalability

As shown in section 2.5, verification and identification problems are quite different to solve in terms of accuracy and speed. These two constraints take importance when an identification process in a biometric system is required, such as in civilian or forensic applications, where negative recognitions are needed. In such a process a comparison with any template present in the database in the worst case is computed. Thus if the number *N* of the users enrolled in the database is quite large, the identification process for an individual could take a lot of time (e.g. more than one minute if *N* is about 1 million). Moreover a major number of comparisons leads to an increase of the FAR.

During identification the target is then to reduce the number of comparisons a system has to compute, and this is usually achieved by filtering or indexing processes, that is choosing subsets of users to search in cause the more probability to contain the true identity of the user, using extrinsic or intrinsic factors[1]. For example, in (18) a previous face identification (that is fast) phase that retrieves a small subset of enrolled users is performed, then a second fingerprint identification phase (reliable) is computed on that subset.

Improving the identification phase in a biometric system is still an active area of research.

2.6.3. Security and Privacy

Although it is difficult to steal someone's biometric traits, it is still possible for an impostor to circumvent a biometric system in a number of ways and at different levels (Fig 16). For a detailed description of the several types of attacks refer to (19).

Recent researches have shown that an attacker can lift and replicate the biometric traits, which later can be used to attack on biometric systems (20) (21) (22). For example it is possible to construct fake or spoof fingers using lifted fingerprint impressions. Systems where all the modules (Sensor, feature extractor, matcher and template) are in a smart card, known as match-on-card or system-on-card technologies, have shown they are not the best way to secure a biometric environment, since they are not appropriate for large-scale verification and user has to carry the card with him all the times.

While securing a system, it is also important to pay attention to the templates stored in the database. Stealing a template allows an impostor to replay it to the matcher to gain unauthorized access, and moreover it is possible for him to create a physical spoof from the template (23) to gain unauthorized access to other systems which use the same biometric trait as well.

Moreover a genuine user's characteristic can be covertly acquired by an impostor just being in the physical proximity of the person he is attempting to impersonate. The easiest example is getting

[1] Extrinsic factors could be gender, age, ethnicity, while intrinsic factors could be fingerprint pattern class

the fingerprint from a surface touched by the user, and thus allowing the impostor to spoof attack the system even without having access to the biometric template.

Another problem when a biometric trait is stolen, is that the genuine user subjected to the theft can't easily switch to another set of uncompromised identifiers, like in knowledge-based systems (where you just change password). This problem makes him an unreliable user because the system could allow an impostor to enter believing him as the genuine user.

Privacy concerns are related to the fact that public feels biometric authentication as an intrusive process and they don't know how the information collected will be used at a later time. This because *function creeps*[1] on the stored data can occur. (24)
For example, Disney World in Orlando collects fingerprints from park visitors in order to prevent customers from sharing the tickets with others (25). However, it is possible that the same fingerprints may be used later for searching against a criminal fingerprint database or cross-link it to a person's health records.

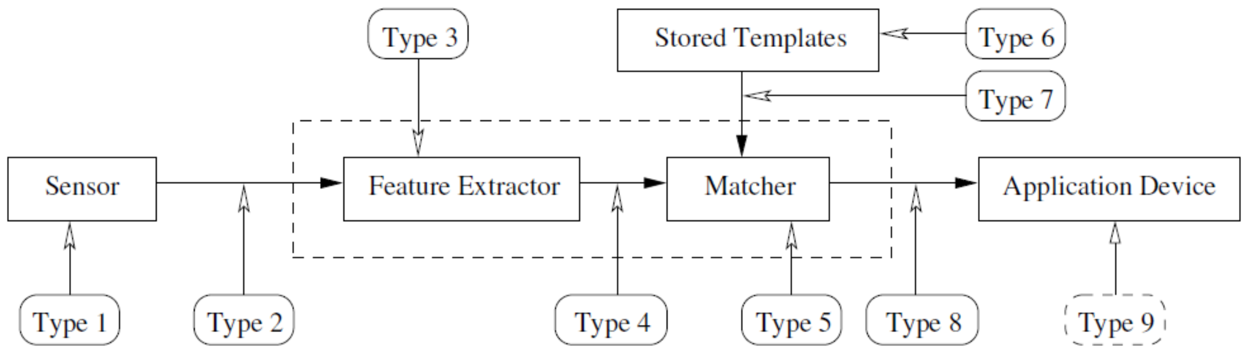


Figure 16 : Biometric System and the nine different points of attack

[1] use for applications beyond their original purpose

3. Multiobiometrics

In section 2.6.1 we talked about problems that affect a biometric systems, and in section 2.6.3 we talked about the problem of spoof attacks to such a type of systems, referring to the case where a single user's trait is used as way to identify him. These problems can be solved, or at least made less dangerous, by the *Multibiometrics*, that are also referred as *"Multimodal Biometrics"* (Fig.17) . Multimodal biometric systems are those which utilize, or are capability of utilizing, more than one physiological or behavioral characteristic for enrollment, verification, and/or identification. Such systems are expected to be more reliable due to the presence of multiple, independent pieces of evidence. The independence is fundamental cause allows making probability assumptions on using match scores density functions coming from the several matchers (26).

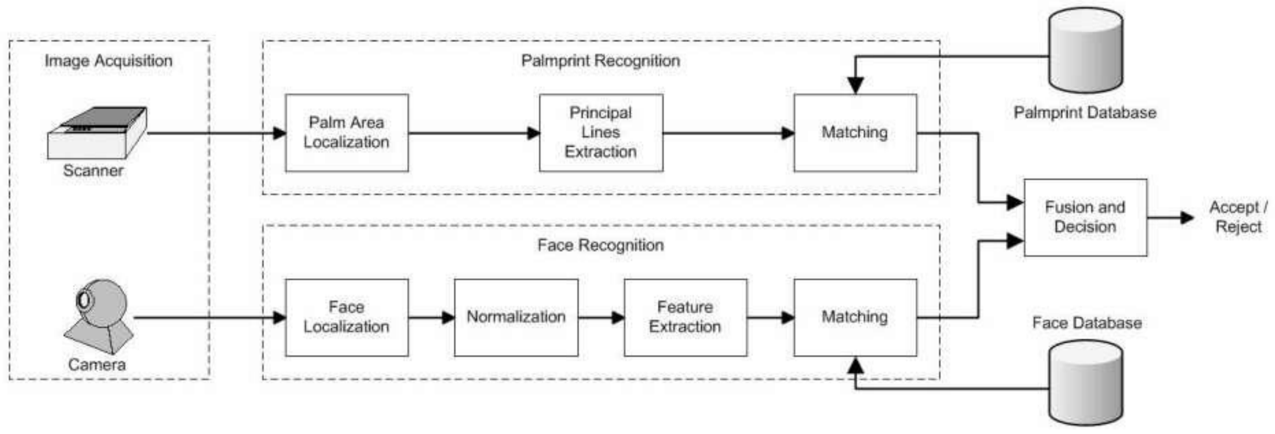


Figure 17 : An example of a multibiometric system in verification mode using palmprint and face traits.

3.1. Advantages

Multimodal biometric systems address the following issues :

- Combining the evidence obtained from different sources using an effective fusion scheme can significantly improve the overall accuracy of the biometric system. The presence of multiple sources also effectively increases the dimensionality of the feature space and reduces the overlap between the feature spaces of different individuals.
- Multibiometric systems can address the non-universality problem and reduce the FTER. For example, if a person cannot be enrolled in a fingerprint system due to worn-out ridge details, he can still be identified using other biometric traits like face or iris. Subsequently during authentication the application may ask only a subset of the biometric characteristics, avoiding the user to provide what he can't.
- Multibiometric systems can also provide a certain degree of flexibility in user authentication. Suppose a user enrolls into the system using several different traits. Later,

at the time of authentication, only a subset of these traits may be acquired based on the nature of the application and the convenience of the user. For example, consider a banking application where the user enrolls into the system using face, voice and fingerprint. During authentication, the user can select which trait to present depending on his convenience. While the user can choose face or voice modality when he is attempting to access the application from his mobile phone equipped with a digital camera, he can choose the fingerprint modality when accessing the same application from a public ATM or a network computer.

- The availability of multiple sources of information considerably reduces the effect of noisy data. If the biometric sample obtained from one of the sources is not of sufficient quality during a particular acquisition, the samples from other sources may still provide sufficient discriminatory information to enable reliable decision-making.
- Multibiometric systems can provide the capability to search a large database in a computationally efficient manner. This can be achieved by first using a relatively simple but less accurate modality to prune the database before using the more complex and accurate modality on the remaining data to perform the final identification task. This will improve the throughput of a biometric identification system.
- Multibiometric systems are more resistant to spoof attacks because it is difficult to simultaneously spoof multiple biometric sources. Further, a multibiometric system can easily incorporate a challenge-response mechanism during biometric acquisition by acquiring a subset of the traits in some random order (e.g, left index finger followed by face and then right index finger). Such a mechanism will ensure that the system is interacting with a live user. Further, it is also possible to improve the template security by combining the feature sets from different biometric sources using an appropriate fusion scheme.
- Multibiometric systems can also be viewed as fault tolerant systems, which continue operating even when some biometric sources are no longer reliable, due to sensor or software problems, or due to deliberate tempering by users. This fault tolerance feature is particularly useful in large scale authentication systems that work with a large number of users (e.g. a checking shipments application).

Multibiometric systems don’t present only advantages. On the other side they are more expensive and require more resources for computation and storage than unibiometric systems. Moreover such systems generally require additional time for enrollment, causing some inconvenience to the user. Finally, the accuracy of a multibiometric system can actually be lower than that of the unibiometric system if an appropriate technique is not followed for combining the evidence provided by the different sources.

3.2. Multiobiometric Taxonomy

A variety of different types of multibiometric systems has been developed. They differentiate each other by the fact that they use different sources of evidence (Fig. 18). The most known and widely diffused are :

- Multiple Sensors : they use more than one sensor to capture a single user's biometric trait. For example different cameras (2D cameras) can be used to get an individual's face image and make a 3D representation of it. Another example is the use of optical or capacitance sensors in fingerprint recognition.
- Multiple algorithms (or multiple representations) : several feature extraction algorithms are used to compute the same user's characteristic, leading to a better subsequently matching (e.g., texture and minutiae-based fingerprint matchers). Such a system can be less expensive than different multibiometric systems, but on the other hand, adding new extraction modules can burden the overall computational complexity. An example of multiple algorithms biometric system is reported in (27)
- Multiple instances : several instances of the same type of user's trait are collected and stored in the database. The fingerprints of all the user's fingers could be acquired in a fingerprint system recognition[1], or the iris images of both the user's eyes in a iris system recognition.
- Multiple samples : A single sensor can be used to acquire several samples of the same user's biometric trait, in order to take into account variations in different capture sessions or to gain a more complete description of it. In a face recognition system for example, different positions and right and left profile as well as frontal profile can be saved, making the system ready to face with intra-class variations when will occur. The critical issue in a multiple samples system is to decide the number of samples the user has to give during enrollment. This because the samples will be adequate to represent the variability and typicality of the biometric data.
- Multiple biometric traits : several physiological characteristics are implemented and used to discriminate individuals. A large variety of such systems have been developed, for example combining face and fingerprint (28), or face and iris (29), or fingerprints and iris (30). The choice is often made toward traits that are considered more independent (face, palmprint, iris) than others (speech and lips movement). The critical issue is the cost due to development of several interfaces and to the purchase of several sensors. Another issue is the user-acceptance since that more time is requested to enroll and verify him rather than in an unimodal system.

In the first four scenarios, multiple sources of information are derived from the same biometric trait. In the fiveth scenario, information is derived from different biometric traits and these systems are known as *multimodal* biometric systems. In fact, biometric fusion can also be carried

[1] The FBI IAFIS system combines information from all the ten fingers to identify an user

out on any arbitrary combination of the above five sources and such systems can be referred to as *hybrid* multibiometric systems .

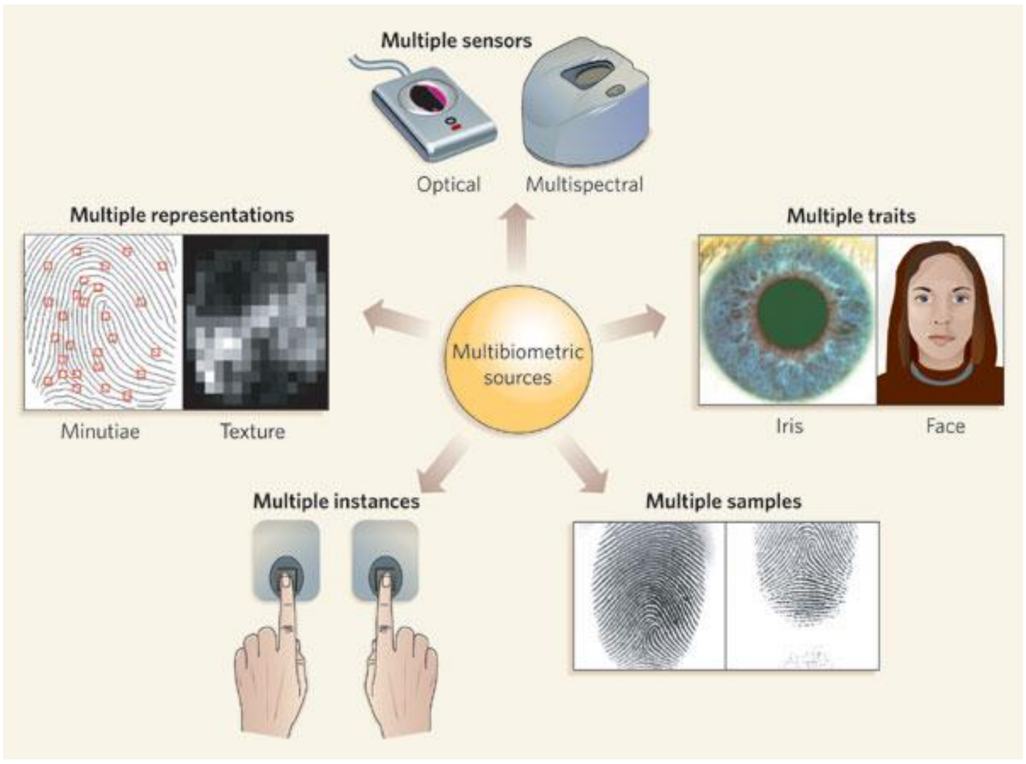


Figure 18 : Various sources of information that can be fused in a multibiometric system. In four of the five scenarios (multiple sensors, representations, instances and samples), multiple sources of information are derived from the same biometric trait. In the fifth scenario, information is derived from different biometric traits and such systems are known as multimodal biometric systems.

3.3. Levels of fusion

In section 3.2 we have seen that in a multibiometric system we need to take some design choices:

- Sensors to use.
- Which algorithms to use for representation (feature extractor) and matching of data coming from the sensors.
- How many samples of the same biometric trait need to be acquired.
- Which instances of the same class of characteristic (which fingers for fingerprint).
- Which biometric traits should be used in the system.

Other major issues that need to be considered are the sequence in which information sources could be acquired from the user, and at which point of the authentication phase (in section 2.1 it has been described that the unimodal biometric system authentication process follows the scheme presented in Figure 19) these information need to be fused[1] in order to get a final decision regarding genuine or impostor nature of the individual.

Furthermore another issue is represented by the cost-performance (FAR-FRR) trade-off. The choice must to be taken basing on the application itself and basing on the context it will work in.

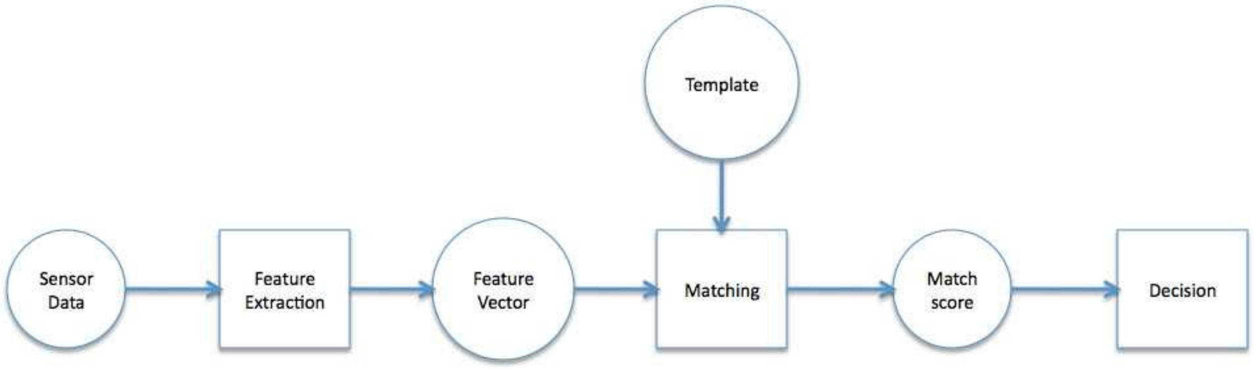


Figure 19 : Stream data flow of the authentication process

In a multibiometric system the sequence in which the multiple sources of information are acquired and processed can be serial mode, parallel mode, or hierarchical mode. In the serial mode, the output of one biometric trait is typically used to narrow down the number of possible identities before the next trait is used. This serves as an indexing scheme in an identification system. For example, a multimodal biometric system using face and fingerprints could first employ face information to retrieve the top few matches, and then use fingerprint information to converge onto a single identity. This is in contrast to a parallel mode of operation where information from multiple traits is used simultaneously to perform recognition. This difference is crucial. In the serial (or cascade) operational mode, the various biometric characteristics do not have to be acquired simultaneously. Further, a decision could be arrived at without acquiring all the traits (Fig. 20). This reduces the overall recognition time. In the hierarchical scheme, individual classifiers are combined in a treelike structure. In the case of biometric acquisition, both serial and parallel architectures are quite common. It is usually convenient and cost-effective to acquire physically related biometric traits simultaneously. For example, face, voice and lip movement can be simultaneously acquired using a video camera.

Not just the sequence in which the multiple sources are acquired but even their sequence processing can be done in parallel or cascade mode. In the former case the primary goal of system designers has been a reduction in the error rate of biometric systems. The parallel mode generally has a higher accuracy because it utilizes more user's evidence for the recognition. In the latter case a cascading architecture may have other advantages such as increased user convenience and higher throughput, which may be useful in large scale identification tasks. . In (31) a cascade scheme is used just for that reason: the user does not have to provide all the biometrics every time.

[1] The process of integrating evidence provided by different biometric sources is known as biometric fusion

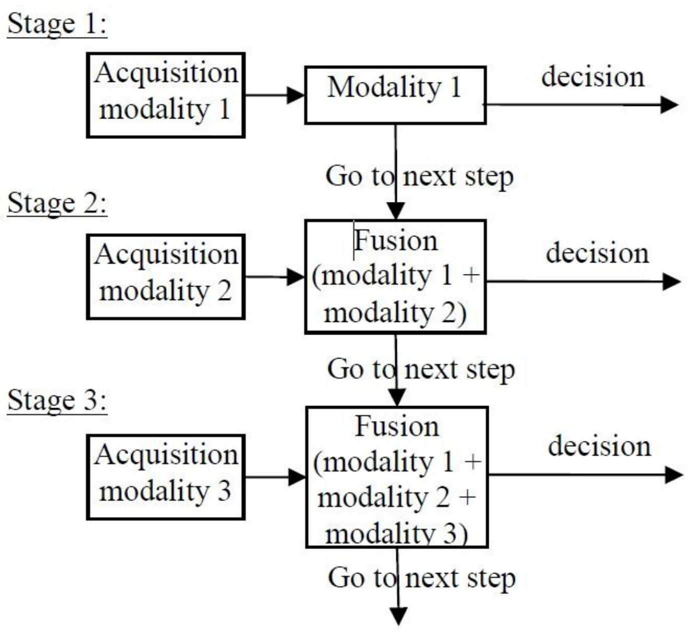


Figure 20 : Example of serial/cascade acquisition mode. Here the next step is performed if necessary, basing on the previous decision

From the literature it comes out that a multibiometric system designer usually can face the fusion problem choosing between five different level points where apply the data fusion of different biometric sources (32):

- Fusion at *Sensor Level*
- Fusion at *Feature Extraction Level*
- Fusion at *Matching Score Level*
- Fusion at *Decision Level*
- Fusion at *Rank Level*

First of all, such fusion schemes can be divided into two major groups (33): pre-classification or fusion be*fore* matching, and post-classification or fusion *after* matching. Such a categorization is necessary since the amount of information available for fusion reduces drastically once the matcher has been invoked. Pre-classification fusion schemes typically require the development of new matching techniques (since the matchers used by the individual sources may no longer be relevant) thereby introducing additional challenges. Pre-classification schemes include fusion at the sensor (or raw data) and at the feature extraction level while post-classification schemes include fusion at the match score, rank and decision levels.

3.3.1. Fusion at Sensor Level

The raw biometric data (e.g., a face image) acquired from an individual represents the richest source of information although it is expected to be contaminated by noise (e.g., non-uniform illumination, background clutter, etc.). Sensor level fusion refers to the consolidation of (a) raw data obtained using multiple sensors, or (b) multiple snapshots of a biometric using a single sensor. For example in face authentication, 2D texture information coming from a camera and 3D data coming from different sensors (depth) could be fused to make a 3D texture face image, subsequently used in the feature extraction process.

3.3.2. Fusion at Feature extraction Level

In feature-level fusion, the feature sets originating from multiple biometric algorithms are consolidated into a single feature set by the application of appropriate feature normalization, transformation and reduction schemes (Fig. 21). The primary benefit of feature-level fusion is the detection of correlated feature values generated by different biometric algorithms and, in the process, identifying a salient set of features that can improve recognition accuracy. Eliciting this feature set typically requires the use of dimensionality reduction methods and, therefore, feature-level fusion assumes the availability of a large number of training data.

The fused feature vector originated during this stage is then compared for matching with a template that has been created after the fusion of the different features vector as well, otherwise it would not make sense computing similarity/distance scores[1] between uncorrelated vectors.

Two major problems have been identified in this fusion level :

- The feature vectors that should be combined may be incompatible (due to size factors for instance), or due to the fact that just one biometric trait data could not be available cause user's malformations. While the former could be solved designing a specific system (padding the vectors for example), the latter would lead to a FTE (Failure To Enroll) case.
- The size of features vectors extracted from the raw data may be very large, and could end in a very expensive matching process when comparing the feature vector with the template (the situation would be much worse in identification mode), since even in unimodal systems this phase requires lot of computational resources.

[1] Scores after matching could be similarity (how much two templates are correlated) or distance (how much two templates differ) scores.

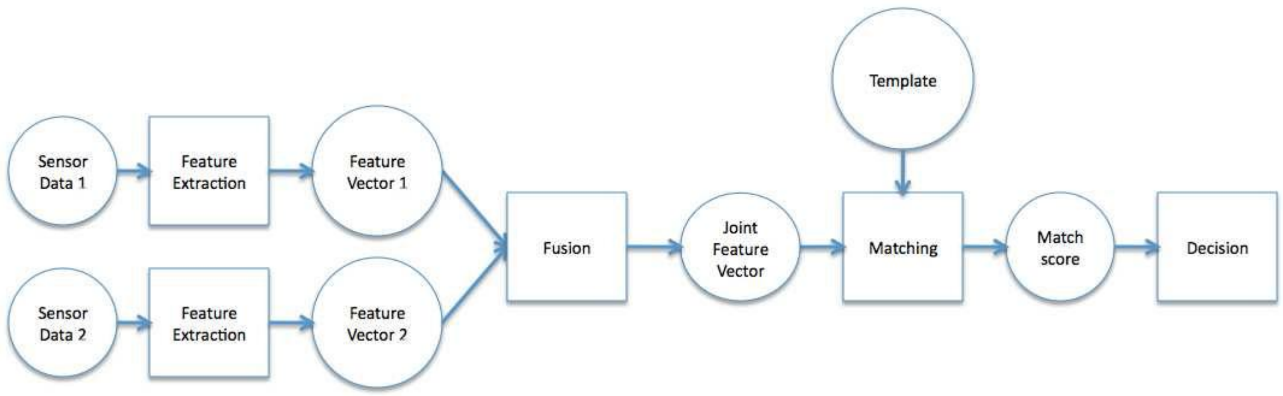


Figure 21 : Fusion at Feature Extraction Level in a multibiometric system

3.3.3. Fusion at Matching Score Level

In score-level fusion the match scores output by multiple biometric matchers are combined to generate a new match score (a scalar) that can be subsequently used by the verification or identification modules for rendering an identity decision. Fusion at this level is the most commonly discussed approach in the biometric literature primarily due to the ease of accessing and processing match scores (compared to the raw biometric data or the feature set extracted from the data). Fusion methods at this level can be broadly classified into three categories :

- Density-based schemes : This approach is based on the likelihood ratio test and it requires explicit estimation of genuine and impostor match score densities. The density based approach has the advantage that it directly achieves optimal performance at any desired operating point (FAR), assuming that the score densities can be estimated accurately. In fact, a comparison of eight biometric fusion techniques conducted by (34) with data from 187,000 subjects concluded that “Product of Likelihood Ratios was consistently most accurate, but most complex to implement” and “complexity in this implementation is in the modeling of distributions, rather than fusion per se”. The statement in (34) about the complexity of density estimation was based on the use of kernel density estimator (KDE). The selection of kernel bandwidth and density estimation at the tails proved to be the most complex steps. Using density-based schemes in fact requires to estimate matching score conditional densities, because they don’t usually follow well known distributions (e.g. Gaussian distribution).
- Transformation-based schemes : The match scores are first normalized (transformed) to a common domain and then combined using product, sum, max or min rules [108]. Choice of the normalization scheme and combination weights is data-dependent and requires extensive empirical evaluation
- Classifier based schemes : Scores from multiple matchers are treated as a feature vector and a classifier is constructed to discriminate genuine and impostor scores [15,66,127]. When biometric score fusion is considered as a classification problem, the following issues pose challenges. (i) Unbalanced training set: The number of genuine match scores available

for training is $O(N)$, but the number of impostor scores is $O(N^2)$, where N is the number of users in the database. (ii) Cost of misclassification: Depending on the biometric application, the cost of accepting an impostor may be very different from the cost of rejecting a genuine user. For example, a biometric system deployed in a security application typically is required to have a false accept rate (FAR) of less than 0.1%. Therefore, the fusion strategy needs to minimize the false reject rate (FRR) at the specified FAR values rather than minimizing the total error rate (sum of FAR and FRR). (iii) Choice of classifier: Given a variety of admissible classifiers, selecting and training a classifier that gives the optimal performance (minimum FRR at a specified FAR) on a given data set is not easy.

In Matching score fusion level the feature vectors are computed independently using the raw data coming from the sensors, and later compared with related templates present in the user's database, which in turn are stored separately according with their own biometric trait. A similarity/distance score is then retrieved for each of them and then fused in a single score (Fig. 22).

It must be noted that the match scores generated by the individual matchers may not be homogeneous. For example, one matcher may output a distance or dissimilarity measure (a smaller distance indicates a better match) while another may output a similarity measure (a larger similarity value indicates a better match). Furthermore, the outputs of the individual matchers need not be on the same numerical scale (range). Finally, the match scores may follow different probability distributions and may be correlated. These factors make match score level fusion a challenging problem, and rise the need of a normalization technique used to put the several scores in a common domain[1].

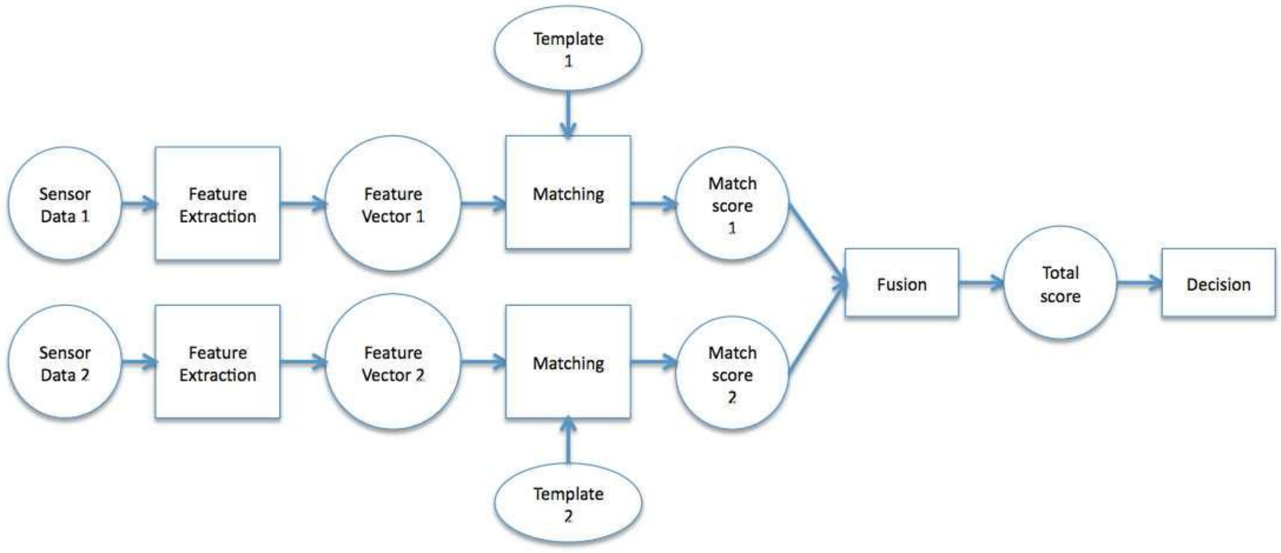


Figure 22 : Fusion at Match-Score Level in a multibiometric system

[1] The discussion about Score Normalization Techniques will be faced in the following paragraph

The most used fusion strategies are :

- Sum Rule : the simplest way of combination is computing the sum of the scores and apply a threshold to make a decision. Usually weights are computed and a weighted sum is then calculated:

$$S = \sum_{j=1}^{n} (W_j \times S_j)$$

  Where S_j is the match score of j^{th} trait and W_j is the weight assigned to the j^{th} trait and $\sum_{j=1}^{n} W_j = 1$
  Two types of weights are usually assigned:
    - Matcher weights : every matcher is assigned a weight according to some information regarding for example the performance of the matcher itself. In a multi-algorithm system different matchers regarding the same trait would usually have different weights.
    - User weights : every user has his own weigths for each biometric trait (and in turn for each matcher related to that trait)[1].
- Support Vector Machines (SVM): The main idea of SVM is to separate the training data into two classes with a hyperplane that maximizes the margin between them. In (35) they used a two-class classification approach, in particular the C-support vector classification (C-SVC) formulation.
- Decision Tree : A decision tree derives a sequence of if–then–else rules using the training set in order to assign a class label to the input data. It does this by finding out an attribute (feature) that maximizes information gain at a particular node. The C5.0 program Quinlan (1992) was used to generate a tree from the training set of genuine and impostor score vectors. The training set consisted of 11,125 impostor score vectors and 250 genuine score vectors. The test set consisted of the same number of impostor and genuine score vectors. Cross-validation was done by considering 10 such partitions of the training and test sets. Figure 23 shows the construction of the C5.0 decision tree on one such training and test set.
- Linear Discriminant Analysis (LDA) : Linear discriminant analysis of the training set helps in transforming the n-dimensional score vectors into a new subspace that maximizes the between-class separation. In (36) LDA is used (along with Decsion tree and SVM) and the plot of the score vectors using the first and the second discriminant variables is shown. The test set vectors are classified by using the minimum Mahalanobis distance rule (after first calculating the centroids of the two classes in the new feature space, and then measuring the Mahalanobis distance). We assume that the two classes have unequal covariance matrices.

[1] See section

In the literature seems that the weighted sum generally performs better than the other two decision tree and LDA methods, even if sometimes this is not true as reported in (37) and (38) where the SVM trained classifier outperform the mean or weighted mean rule. In (39) they compared Arithmetic Mean Rule (AMR) with SVM methods. They stated there is no one better than the other, but depends on the genuine and impostor score distributions, in particular on the overlap area among them: if the two classes are easy to discriminate then the AMR with a "standard" normalization is preferred, otherwise when the overlap of both distributions is important the SVM outperforms the AMR.

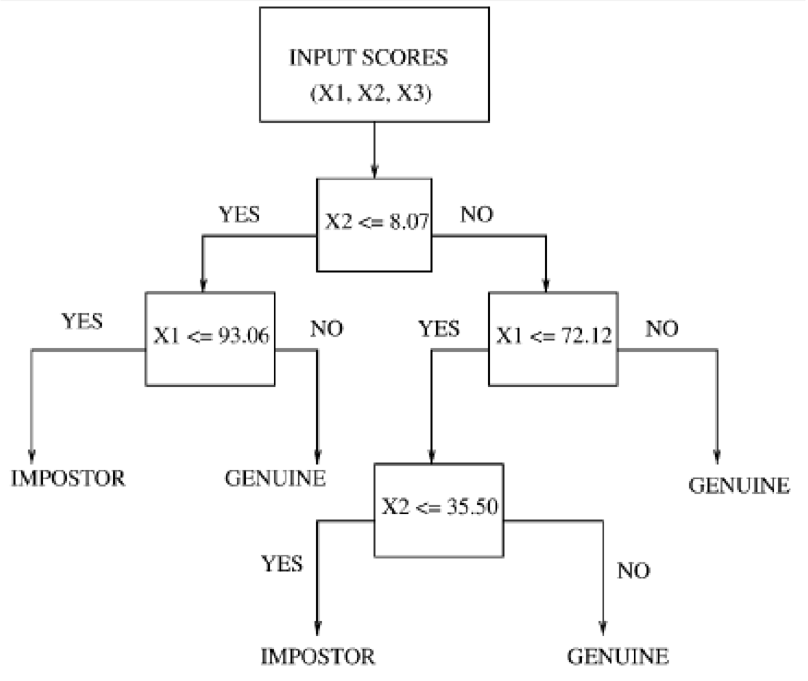


Figure 23 : Decision tree example

3.3.4. Fusion at Decision Level

In a multibiometric system, fusion is carried out at the *abstract* or *decision* level when only the decisions output by the individual biometric matchers are available. Many commercial off-the-shelf (COTS) biometric matchers provide access only to the final recognition decision. When such COTS matchers are used to build a multibiometric system, only decision level fusion is feasible. Methods proposed in the literature for decision level fusion include:

- AND rule : it is a well known and widely used rule, that needs all the matchers to give positive outputs at the same time to allow a positive final decision. As the intuition suggests, the drawback of such a strategy is that the FRR will increase significantly, while the FAR will be kept low.
- OR rule: another well known strategy, where it is sufficient that only a biometric trait is recognized as genuine to lead in a positive final decision. Such a fusion rule allows to develop a multibiometric system where the serial matching approach is used with the

possibility of not acquiring all the traits. This strategy instead decreases the FRR of the system, but on the other hand increases the its FAR. Moreover, the FAR could be even more than in a unimodal system. This is due to the fact that multiple biometric sources are available, thus more probability than at least one of them is similar to the ones of the user the impostor is claiming to be.

- Majority Voting rule : as reported in (40) it seems, under some assumptions, to be the best fusion method at this level. It takes a positive decision if the majority of the outputs from multiple matchers were positive, negative otherwise.

Other fusion methods are the Weighted Majority Voting (41), Bayesian decision fusion and Dempster-Shafer theory of evidence (42).

Fusion at this level means that the system works as if each subsystem would be independent from the others and reaches its own decision about the genuine or impostor nature of the user. Only at this point the information are fused in a single and final decision, since no more modules in the system are present (Fig. 24).

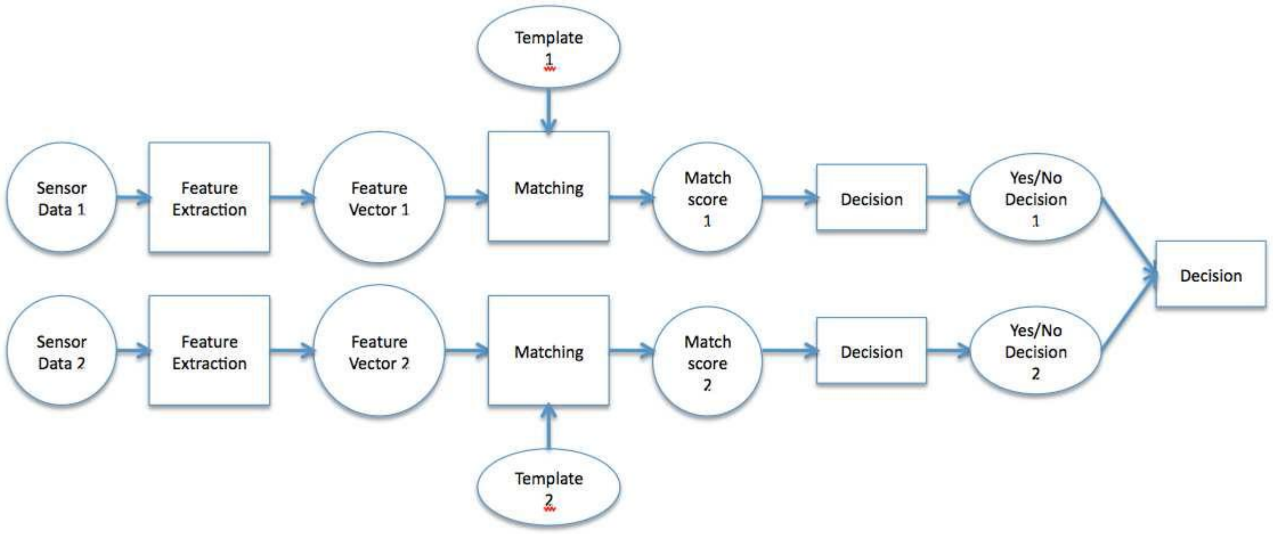


Figure 24 : Fusion at Decision Level in a multibiometric system

3.3.5. Fusion at Rank Level

This type of fusion is relevant in identification systems where each classifier associates a rank with every enrolled identity (a higher rank indicating a good match). Thus, fusion entails consolidating the multiple ranks associated with an identity and determining a new rank that would aid in establishing the final decision. Techniques such as the Borda count may be used to make the final decision.

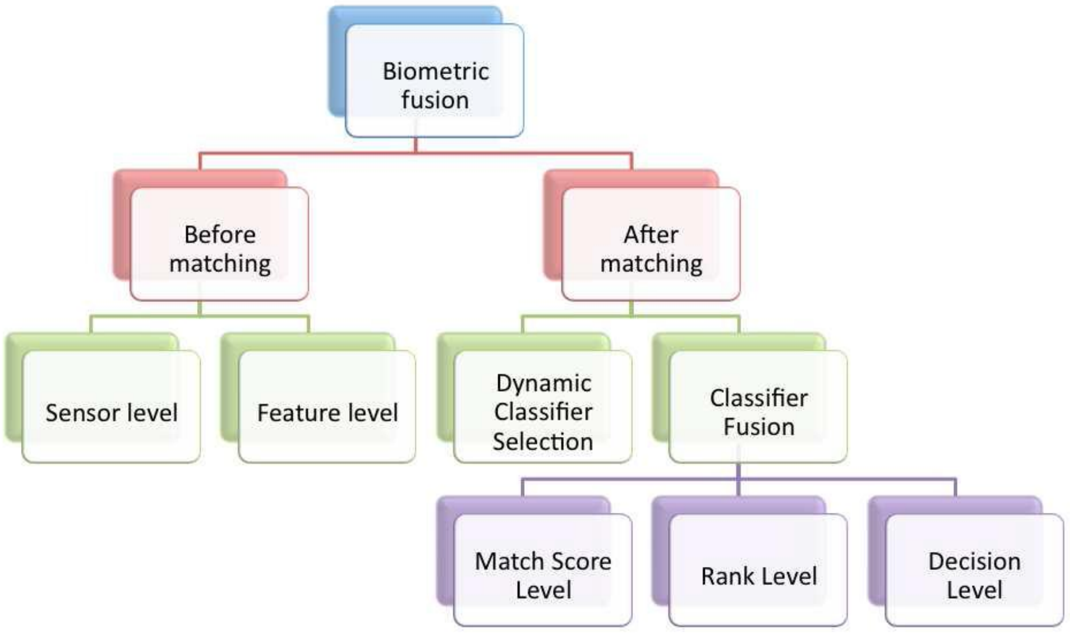


Figure 25 : Summary diagram of possible levels of information fusion

3.4. Score normalization

In the previous section it has been noted that at various points of the multibiometric system data flow the information fusion can be made. Often the matching score level is preferred since it presents a good trade-off between information richness and easiness on computing it.

Since different matching modules outputs matching scores in different domains, a common one is needed in order to allow subsequent fusion computing. Moreover, some matchers could output similarity scores of the compared templates, instead others could outputs distance scores. It is evident that fusing similarity and distance scores together without normalizing them has no sense. Distance scores can be transformed into similarity scores by subtracting the normalized score from 1. Another factor that could aggravate the situation is the different performances of different matchers, as a fingerprint matching module generally performs better than a face recognition module.

As an example, in (43) face, fingerprint and hand-geometry biometric sources are acquired and different scores in range and similarity/distance domains are computed, showing the need of a normalization technique.

Normalization techniques refers to changing the location and scale parameters of the matching score distributions so that they are transformed in a common domain. When the parameters used for normalization are determined using a fixed training set, it is referred to as *fixed score normalization*.

A normalization technique is said to be a good if it estimates location and scale parameters in a robust and efficient way. Robustness refers to insensitivity to the presence of outliers. Efficiency refers to the proximity of the obtained estimate to the optimal one when the distribution of the data is known.

These are several quick descriptions of some normalization schemes :

- Min-Max normalization : it is the simplest technique. The minimum and maximum values of the scores produced by a matcher need to be known. If not they need to be estimated using a training set of images. But in the latter case we need to take into account that this method is not robust, because it is highly sensitive to outliers in the data used for estimation. Given a set of matching scores $\{s_k\}$, *K* = 1,2,….,*n* the normalized scores are given by

$$s_k^{'} = \frac{s_k - min}{max - min}$$

  Min-Max normalization retains the original distribution of scores except for a scaling factor and transforms all the scores into the range $[0,1]$.

- Decimal scaling normalization : can be applied when the matching scores of different matchers are in logarithmic scale, e.g. if one matcher has values in the $[0,1]$ range and another one in the $[0,100]$ range (scale factor = 10^2). The normalized score is given by :

$$s_k^{'} = \frac{s_k}{10^n}$$

  Where $n = \log_{10} \max(s_i)$. The problem is that the logarithmic scale constraint is much strict leading to lack of robustness.

- Z-score normalization : one of the most commonly used normalization together with the min-max function, it is calculated using the arithmetic mean and standard deviation of the given data, these given by the matcher producer or estimated from a given set of matching scores. The function is the following :

$$s_k^{'} = \frac{s_k - \text{}}{\sigma}$$

  Where is the arithmetic mean and σ is the standard deviation of the given data. Three drawbacks come to the attention : (i) both mean and standard deviation are sensible to outliers (ii) it does not guarantee a common numerical range for the normalized scores of the different matchers (iii) Z-score is optimal if the score distribution follows a Gaussian distribution, otherwise the input distribution is not retained at the output.

- Median and Median Absolute Deviation (MAD) normalization : this normalization scheme is insensitive to outliers and points at the tails of the distribution. The normalized score is given by :

$$s_k^{'} = \frac{s_k - median}{MAD}$$

Where $MAD = median(|s_k - median|)$. But *median* and MAD estimators have lower efficiency compared with mean and standard deviation.

- Tanh-estimators normalization : this scheme is said to be robust and highly efficient whose function is :

$$s_k^{'} = \frac{1}{2}\left\{\tanh\left(0.01\left(\frac{s_k - \mu_{GH}}{\sigma_{GH}}\right)\right) + 1\right\}$$

Where μ_{GH} and σ_{GH} are the mean and standard deviation estimates, respectively, of the genuine score distribution as given by the Hampel estimators based on the following influence (ψ)-function :

$$\psi(u) = \begin{cases} u & 0 \leq |u| < a \\ a\, sign(u) & a \leq |u| < a \\ a\, sign(u)\left(\frac{c - |u|}{c - b}\right) & b \leq |u| < c \\ 0 & |u| \geq c \end{cases}$$

a,b and c are bounds where a certain amount of scores can be found there. For example in (43) a, b and c were chosen such that 70% of the scores where in the interval $(m - a, m + a)$, 85% of the scores where in the interval $(m - b, m + b)$ and 95% of the scores where in the interval $(m - c, m + c)$, where *m* is the median score. In other words a, b and c represents the points at the tails of the distribution. This method is not sensitive to outliers since their influence is reduced. In the case of, however, a large number of outliers is present, the estimate is robust since they not influence it, but not efficient. On the other hand, if many tail-points influence the estimate, the estimate is not robust but the efficiency increases. Therefore a, b and c parameters must be carefully chosen regarding the amount of robustness required, which in turn depends on the estimate of the amount of noise in the training data.

In (44) another normalization scheme, Adaptive Normalization, is proposed. It is based on the idea that the errors of individual biometric matchers stem from the overlap of the genuine and impostor score distributions. The overlap region is characterized by its center *c* and its width *w*. To decrease the effect of this overlap on the fusion algorithm, they proposed to use an adaptive normalization procedure that aims to increase the separation of the genuine and impostor distributions, while still mapping the scores to [0,1] range.

In (35) a new normalization method is proposed. The presence of outliers in the dataset reduces separation degree between genuine and impostor scores. The Reduction of High-Scores Effect (RHE) normalization is derived from the min-max normalization and inherits its simplicity and retains the original distribution of genuine scores except for a scaling factor, as well as distributing them more evenly. The given function for calculating the normalized score is :

$$x' = \frac{x - \min(X)}{\{mean(X^*) + std(X^*)\} - \min(X)}$$

Where X denotes the distribution of all raw scores (genuine and impostor) and X^* denotes the distribution of genuine scores. Mean and standard deviation of only genuine scores have been used due to the idea that a multimodal biometric system suffer mainly from the “low” genuine scores instead of “high” impostor scores. It is in fact easier that a genuine produces a low score (due to factors such as change of the biometric trait, alteration of condition between enrollment and testing phase, the level of subject’s cooperation etc.) rather than an impostor produces a high score.

3.5. State of the Art

3.5.1. Generalized densities

In section 3.3.3 the distribution-based fusion technique at matching score level has been introduced. In (45) they proposed an evaluation of score fusion methods just based on densities of genuine and impostor scores. In (46) a theoretical framework based on posteriori probability inferred using Bayesian theory has been studied. The posteriori probability refers to the probability which a pattern *Z* should be assigned to class ω_j , *j* = 1, ….m as final combined decision given the measurement vectors $\vec{x}_i$, with *i* = 1, …., *R,* where $\vec{x}_i$ is the vector used by the *i*th classifier, *R* is the number of the classifiers and *m* is the number of possible classes.
Z is assigned to the class ω_j if :

$$P(\omega_j | \vec{x}_1, \ldots, \vec{x}_R) = \max_k P(\omega_k | \vec{x}_1, \ldots, \vec{x}_R)$$

Where

$$P(\omega_k | \vec{x}_1, \ldots, \vec{x}_R) = \frac{p(\vec{x}_1, \ldots, \vec{x}_R | \omega_k) P(\omega_k)}{p(\vec{x}_1, \ldots, \vec{x}_R)}$$

Where in turn $p(\vec{x}_1, \ldots, \vec{x}_R)$ is the unconditional measurement joint probability density:

$$p(\vec{x}_1, \ldots, \vec{x}_R) = \sum_{j=1}^{m} p(\vec{x}_1, \ldots, \vec{x}_R | \omega_j) P(\omega_j)$$

Which suggests that only the $p(\vec{x}_1, \ldots, \vec{x}_R | \omega_j)$ formulation requires to be studied.

$P(\omega_k)$ is the a priori probability of occurrence of class *k* and $p(\vec{x}_1, \ldots., \vec{x}_R|\omega_k)$ is the conditional joint probability density function of the measurements extracted by the R classifiers, given the class ω_k, represented by :

$$p(\vec{x}_1, \ldots., \vec{x}_R|\omega_k) = \prod_{i=1}^{R} p(\vec{x}_i|\omega_k)$$

Where $p(\vec{x}_i|\omega_k)$ is the conditional probability density function of the measurement $\vec{x}_i$ from the *i-th* classifier given the class ω_k. This formulation holds under the assumption that the scores of the different biometric traits are independent.

In the case of multibiometric system only two classes are possible: *G* as genuine and I as impostor. The several classifiers are the biometric trait matchers (face, fingerprint, iris etc.) and the vector $\vec{x}_i$ is just the score s_i obtained from the *i*th matcher after computing the input feature vector related to a biometric trait. This means that, referring to the $p(\vec{x}_i|\omega_k)$ formulation, in general the genuine and impostor matching score distributions can be formulated as :

$$p(s_i|G) \; and \; p(s_i|I)$$

Where s_i is the score obtained from the *i*th matcher (in other words a matcher calculate the score s_i, then this score is normalized by $p(s_i|G)$ which has been previously estimated using training data).

The problem now is how to estimate the genuine and impostor distributions. Three ways seems to be possible :

- In some cases, certain type of distributions are assumed, as in (47) where normal distributions have been used. Usually a Gaussian distribution to represent the matching scores is taken (39). But this approach is possible just in a few cases since often the score distributions don't follow any of the well known function densities.
- The probability is modeled as empirical frequencies of the score s_i obtained by performing the *i*th matcher on a training data. This approach has a practical drawback. In order to get well modeling frequencies of the score probability density function, a very large number of users need to be collected as training data. These must include all the possible heterogeneous aspects of the population and take into account that different biometric traits have different degrees of freedom.
- Various methods of density estimation based on nonparametric techniques have been proposed. In (48) the Gaussian Mixture Model (a specific case of finite mixture models) is used since it usually leads to true density estimates if a sufficient number of training samples are available.
  In (49) the GMM method and Monte Carlo sampling based hypothesis are used to fuse face and speech modalities. GMM is used to estimate mean and standard deviation of the match score distributions, subsequently the Monte Carlo Method is used to sample them in such a manner that the sampled scores will follow a Gaussian distribution. Experimental

results show that the above fusion method achieves consistently higher verification rates as compared with most popular and state-of-the-art fusion schemes such as weighted sum rule and likelihood ratio method on all the five different multimodal biometric databases used.
In (50) they first estimate genuine and impostor generalized densities using a mixture of discrete and continuous components and a Gaussian kernel density estimator. Then they present two approaches for combining evidence based on such generalized densities (i) the product rule (independence assumption) and (ii) copula models (which instead consider the dependence between matching scores of multiple modalities).

The adoption of these methods however, requires a careful choice of some parameters (as histogram bin width or kernel bandwidth) that are critical to the fusion performance.

The motivation of using generalized densities is that some parts of the score distributions can be discrete in nature because some matchers always produce the same predetermined score as output if the number of extracted features is less than a threshold, leading to discrete components ($P(S = s_0) = p > 0$. Thus, estimating the distribution using continuous densities may be inappropriate (since it must be $P(S = s_0) = 0$ for every possible score s_0).

If a good estimate of both the genuine and impostor densities is made, then a widespread fusion scores method is the “likelihood ratio”. Assuming that multiple traits are statistically independent, the corresponding densities are fused using the product rule and thus achieving joint densities. That is : let S_1, S_2,….., S_R, the random variables used to indicate the similarity (dissimilarity) between an input and a template for R different matchers. Let $p_j\left(S_j\middle|genuine\right)$ and $p_j\left(S_j\middle|impostor\right)$, where $j = 1,2,\ldots\ldots,R$, be the conditional probability density functions of the R variables given the genuine and impostor classes respectively. Then the joint conditional probability density function of S_1, S_2,….., S_R given the genuine and impostor classes have the form:

$$p(S_1,S_2,\ldots\ldots,S_R|genuine) = \prod_{j=1}^{R} p_j\left(S_j\middle|genuine\right)$$

$$p(S_1,S_2,\ldots\ldots,S_R|impostor) = \prod_{j=1}^{R} p_j\left(S_j\middle|impostor\right)$$

We call them $f_{gen}\,(\vec{s})$ and $f_{imp}\,(\vec{s})$ where $\vec{s} = [s_1, s_2, \ldots\ldots., s_R]$ is the vector made by the scores coming from the R matchers. Then the likelihood ratio (LR) fusion rule is given by :

$$\vec{s}\,\epsilon \begin{cases} genuine, & if\ \dfrac{f_{gen}\,(\vec{s})}{f_{imp}\,(\vec{s})} \geq \eta \\ impostor, & otherwise \end{cases}$$

Where η is a threshold determined based on the specified FAR. That is, according to the Neyman-Pearson theorem we can select the threshold η such that the likelihood ratio test maximizes the genuine accept rate (GAR) given the specified FAR. However this optimality of likelihood ratio test is guaranteed only when the underlying densities are well estimated.

3.5.2. Quality measures

Quality biometric sample measures have been proposed as well, since it is well known that they have a significant impact on the accuracy of a matcher. When the biometric trait is acquired a quality score of the sample is computed, so that then incorporate it during the fusion process and get a bias of the score coming from that characteristic. The quality learning schemes are, in general, concerned with adjusting the balance of weighting in fusion in favour of the modalities of better quality. Relatively high scores bound with low quality measures are not trustable. In other words, such techniques involve emphasising or deemphasising the scores for the individual biometric modalities during the fusion process, depending on an estimate of their relative degradation.

In (48) a quality vector including quality measures of the match scores is calculated and incorporated into the likelihood ratio framework :

$$QLR(\vec{x},\vec{q}) = \frac{\hat{f}_{gen}\,(\vec{x},\vec{q})}{\hat{f}_{imp}\,(\vec{x},\vec{q})}$$

Where $\vec{x}$ and $\vec{q}$ are the match score and quality of the match score vectors respectively, and $\hat{f}_{gen}$ and $\hat{f}_{imp}$ are the genuine and impostor estimated densities with GMM, respectively.

In (51) the proposed approach involves combining score normalization (UCN), SVM and qualitative-based fusion of Face and Speech biometric traits under clean and degraded data conditions. In the case of clean data condition, the results have shown that the use of SVM with UCN leads to the highest accuracy. But the effectiveness of combined UCN and data relative quality with SVM becomes evident when at least one of the biometric modalities is degraded.

3.5.3. Weighted Sum Rule

In section 3.3.3 the weighted sum based fusion technique has been introduced. The method general refers to the calculation of the sum of the output scores of each matcher, where each of them is coupled with a previously set weight, which could be static or dynamic. As examples in (35) they evaluated the performances of sum rule-based fusion and support vector machines (SVM)-based fusion in a multimodal system based on fingerprint, face and finger vein traits. It came out that in both cases the performance depends on the choice of normalization technique.

Moreover no one is better than the other in absolute sense: the former is less complex to implement while the latter is more precise.
The most common schemes used in sum rule are the user-weighting and the matcher-weighting. In (44) both of them are presented.

As another example, in (39) the Arithmetic Mean Rule (AMR) for the fusion of signature and text-independent speaker verifications is used.

In (52) they presented a Cohort-based normalization of face and fingerprint scores, that is a normalization based on the similarity between not only the claimed identity, but also with the neighbors (cohort) of the claimed identity (which have to be calculated in a prior step for every enrolled user). The scores are later fused with sum or product rules.

3.5.3.1. Matcher Weighting

Regarding the matcher weighting, every matcher is assigned a weight so that each output score of it will be multiplied to that value. Usually, if *M* matchers are present, the weights are calculated in such a way that $\sum_{m=1}^{M} w_m = 1$. The aim is to weaken those scores that come from biometric traits less trustable and on the other side increase the relevance of those that are more reliable. After that the fused score is compared with a threshold. In (44) they propose to assign weights to the individual matchers based on their Equal Error Rates (EER's):

$$w^m = \frac{\left(1/\sum_{m=1}^{M} \frac{1}{e^m}\right)}{e^m}$$

Where e^m is the EER of matcher *m*, with *m* = 1,2,…., M, M is the total number of matchers and w^m is the weight associated to the *m*th matcher's score.

3.5.3.2. User Weighting

Regarding the user weighting (44), they proposed the calculation of user weights based on the *wolf-lamb* concept, that is, users that can be easily imitated (lambs) and users that can successfully imitate some other users (wolves). An extending lambness metric is developed and assigned to every pair of user and matcher (*i*,*m*) :

$$w_i^m = \frac{1}{\sum_{m=1}^{M} d_i^m} \times d_i^m$$

Where w_i^m is the weight associated to the *m*th matcher's score for the user *i*, and d_i^m is a measure of the separation of genuine and impostor distributions for every (*i*, *m*) pair calculated as :

$$d_i^m = \frac{\mu_i^m(gen) - \mu_i^m(imp)}{\sqrt{\left(\sigma_i^m(gen)\right)^2 + \left(\sigma_i^m(imp)\right)^2}}$$

Where $\mu_i^m(gen)$ and $\mu_i^m(imp)$ are the means of genuine and impostor distribution for the pair (i,m) and $\sigma_i^m(gen)$ and $\sigma_i^m(imp)$ are the standard deviations.

The difference between the matcher weighting case is that while in the first the matcher weights are the same for every user, in the user weighting case every user keeps his own weight values for every matcher.

3.6. Summary

It is generally believed that a combination scheme applied as early as possible in the recognition system is more effective.

For example, an integration at the feature level typically results in a better improvement than at the matching score level. This is because the feature representation conveys the richest information compared to the matching score of a matcher, while the abstract labels (decision level) contain the least amount of information about the decision being made. However, it is more difficult to perform a combination at the feature level because the relationship between the feature spaces of different biometric systems may not be known and the feature representations may not be compatible. Further, the multimodal system may not have access to the feature values of individual modalities because of their proprietary nature. In such cases, integrations at the matching score or decision levels are the only options. This is also reflected in the nature of research dedicated to multimodal biometric systems: very few published papers report results on a combination at the feature level.

In (53) some fusion and normalization techniques are compared using face and fingerprint modalities. The results show that the simple sum fusion rule generally performs well over the range of normalization techniques.

However in (54) they evaluated a boosting approach for score level fusion based on the Area Under the Curve Mazimization (AUC). The AUC is the area under the ROC curve (GAR-FAR plot) and a performance metric that is invariant to unequal error cost and unbalanced class sample size. They approved the equality of AdaBoost and Rank-Boost.B in sense of AUC optimization, and applying them in multimodal biometrics the results stated both techniques outperformed SUM rule and reached comparable performance of SVM with linear kernel.

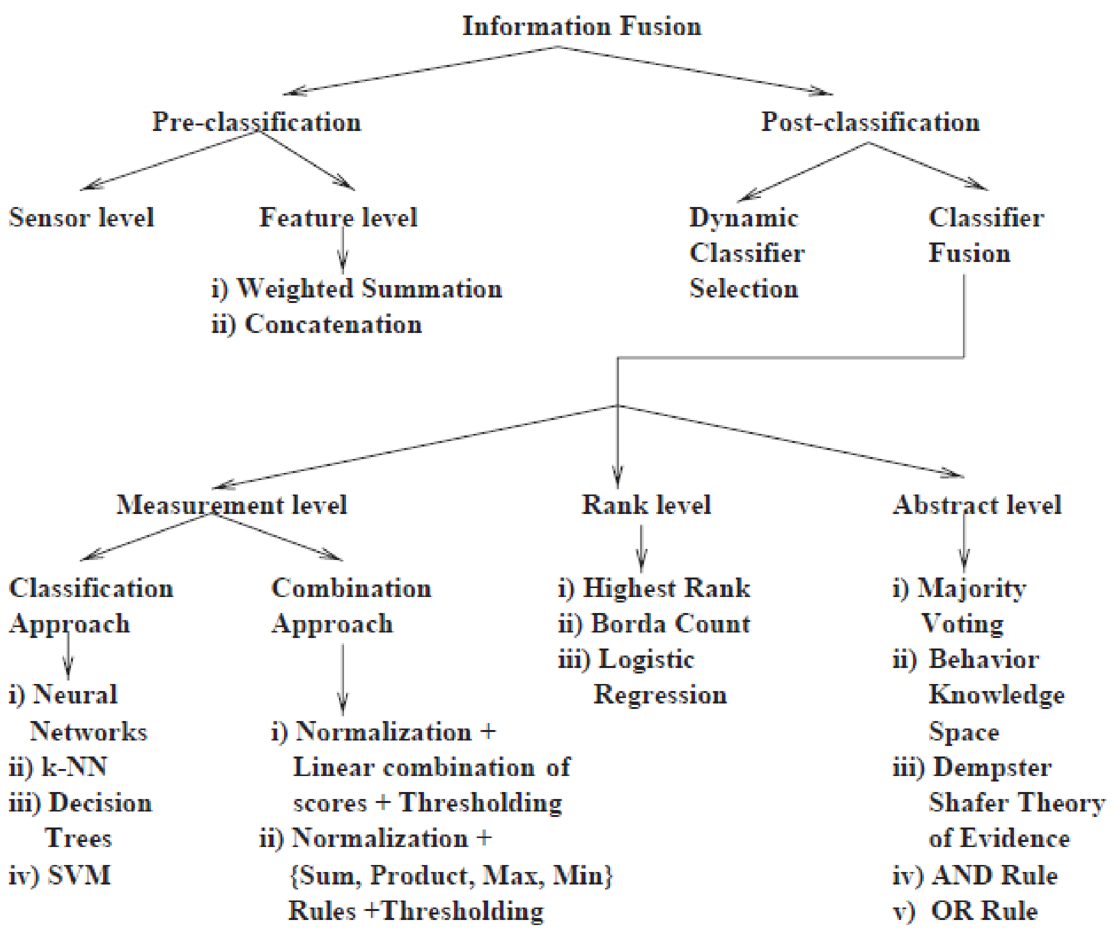


Figure 26 : Approaches to information fusion

4. Deployment

4.1. Face Authentication

Face authentication is essentially a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source. Despite the research on this application started lot years ago[1] and many progress have been achieved, this is still an open field of interest cause it is thought that its performances can be more increased.

Face authentication process splits into two phases, that are Face Detection and Face Recognition.

4.1.1. Face Detection

Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class. Examples include upper torsos, pedestrians, and cars.

Face detection can be regarded as a more general case of face localization. In face localization, the task is to find the locations and sizes of a known number of faces (usually one). In face detection, one does not have this additional information.

Early face-detection algorithms focused on the detection of frontal human faces, whereas newer algorithms attempt to solve the more general and difficult problem of multi-view face detection. That is, the detection of faces that are either rotated along the axis from the face to the observer (in-plane rotation), or rotated along the vertical or left-right axis (out-of-plane rotation), or both. The newer algorithms take into account variations in the image or video by factors such as face appearance, lighting, and pose (55).

The most common used algorithm is known as the “Haar like feature” (56) deriving from the “Haar wavelet” concept, introduced at the early of 20th century. This algorithm works on the feature set based on Haar wavelets rather than pixel intensities of the image (i.e., the RGB pixel values at each and every pixel of image). It has proved that the latter method made the task of feature calculation computationally expensive.

A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums (to do this the original image must to be converted in grayscale). In the detection phase of the Viola–Jones they use three kind of features (Fig. 27) : the value of a *two-rectangle feature* is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent. A *three-rectangle feature*

[1] During 1964 and 1965, Bledsoe, along with Helen Chan and Charles Bisson, worked on using the computer to recognize human faces

computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a *four-rectangle feature* computes the difference between diagonal pairs of rectangles.

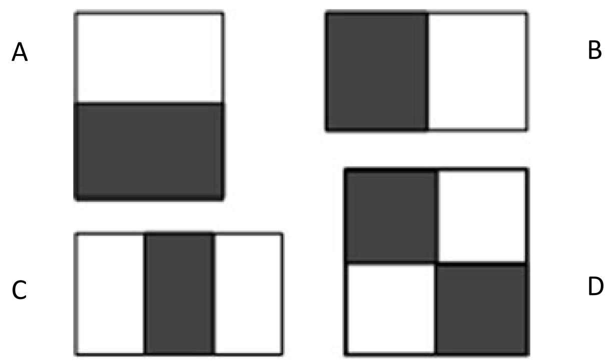


Figure 27 : Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature

The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of *integral images*, a Haar-like feature of any size can be calculated in constant time (approximately 60 microprocessor instructions for a 2-rectangle feature).

The integral image has the same size of the original image and it is derived from it as follows :

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y')$$

Where $ii(x,y)$ is the location of the integral image with coordinates (x,y), and $i(x',y')$ is the pixel value of the original image with coordinates (x',y'). In other words each integral image's location contains the sum of the original image's pixels above and to the left of *x*, *y*. For every location the following two recurrences are sufficient to compute its value :

$$s(x,y) = s(x,y-1) + i(x,y)$$

$$ii(x,y) = ii(x-1,y) + s(x,y)$$

With the overall computation time linear (just one original image scan).

Using the integral image, any rectangular sum can be calculated using four accesses at the integral image locations. In Figure 28 the sum of pixel values within the D rectangle is calculated as P4 + P1 – P2 – P3.

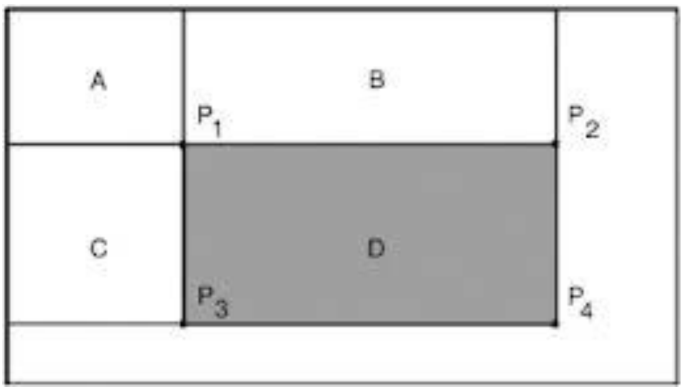


Figure 28 : The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location P1 is the sum of the pixels in rectangle A. The value at location P2 is A + B, at location P3 is A + C and at location P4 is A + B + C + D

Since there are over 180,000 rectangle features associated with each image sub-window it is needed to select the specific Haar features to use along with threshold values. Viola and Jones used a machine-learning method called AdaBoost. AdaBoost combines many "weak" classifiers to create one "strong" classifier. "Weak" here means the classifier only gets the right answer a little more often than random guessing would. That's not very good. But if you had a whole lot of these weak classifiers, and each one "pushed" the final answer a little bit in the right direction, you'd have a strong, combined force for arriving at the correct solution. AdaBoost selects a set of weak classifiers to combine and assigns a weight to each. This weighted combination is the strong classifier. Every such weak classifier is defined as :

$$h_j(x) = \begin{cases} 1 & if\ p_j f_j < p_j \theta_j \\ 0 & otherwise \end{cases}$$

Where f_j is a feature, θ_j is a threshold, $p_j = 1, -1$ is a parity indicating the direction of the inequality sign and *x* is a 24x24 pixel sub-window of an image. The strong classifier is the computed following the next algorithm :

- Given example images $(x_1, y_1), \ldots., (x_n, y_n)$ where $y_i = 0, 1$ fro negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where *m* and *l* are the number of negatives and positives respectively.
- For $t = 1, \ldots., T$:
  - Normalize the weights

    $$w_{i,t} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

    So that w_t is a probability distribution
  - For each feature, *j*, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i - y_i) \right|$
  - Choose the classifier, h_t, with the lowest error ϵ_t
  - Update the weights:

    $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

Where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$

- The final strong classifier is :

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases}$$

where $\alpha_t = \log\frac{1}{\beta_t}$

The acceptance threshold at each level is set low enough to pass all, or nearly all, face examples in the training set. The filters at each level are trained to classify training images that passed all previous stages. (The training set is a large database of faces, maybe a thousand or so.) During use, if any one of these filters fails to pass an image region, that region is immediately classified as "Not Face." When a filter passes an image region, it goes to the next filter in the chain. Image regions that pass through all filters in the chain are classified as "Face." Viola and Jones dubbed this filtering chain a cascade (See Fig. 29).

The order of filters in the cascade is based on the importance weighting that AdaBoost assigns. The more heavily weighted filters come first, to eliminate non-face image regions as quickly as possible. Figure 30 shows the first two features from the original Viola-Jones cascade superimposed on a face image. The first one keys off the cheek area being lighter than the eye region. The second uses the fact that the bridge of the nose is lighter than the eyes (57).

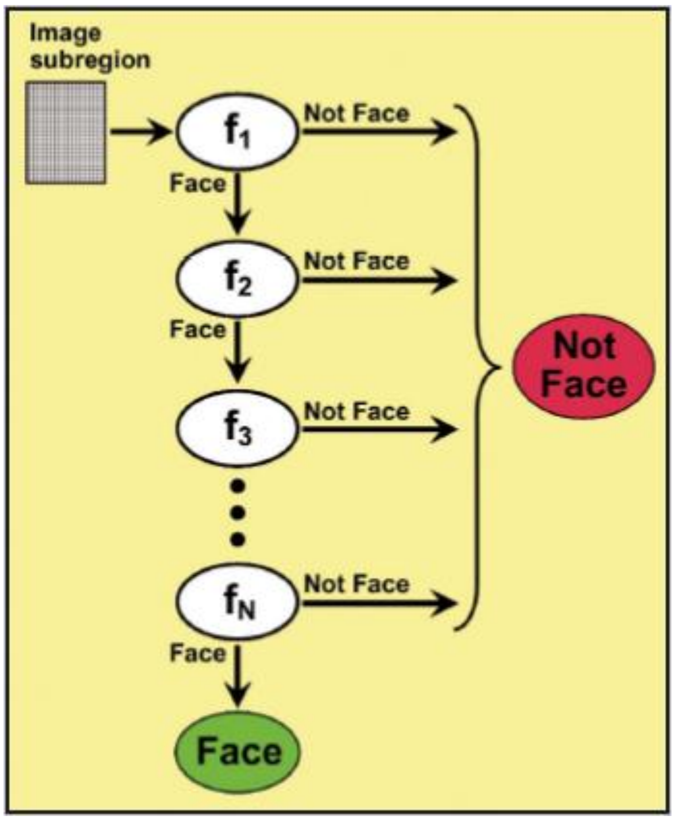


Figure 29 : The classifier cascade is a sequence of weak AdaBoost trained classifiers

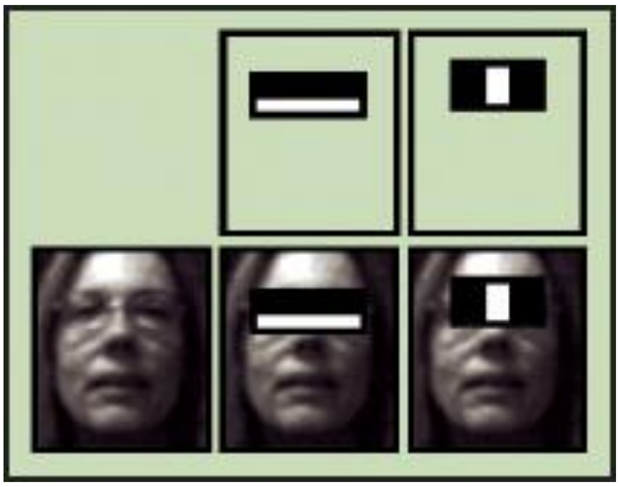


Figure 30 : Two Haar like features selected on an face image

In (58) the face is detected using the triangle based face detection algorithm. Here three face detection algorithms based on RGB, YCbCr and HSI color space models are used. Then ear, eyes and mouth are detected to calculate face boundaries and draw the rectangle on frontal and profile face images. They state that Color processing is much faster than processing other facial features.

4.1.2. Face Recognition

Face recognition refers to the computer process that try to automatically identify or verify a person from a digital image acquired by a photo or a video stream. Usually features from the face, previously detected, are extracted and compared with the features (modeled in a template) present in the database. The use of such a technique is becoming more and more widely used since its overt capability (i.e. the subject does not necessarily know he has been observed) and its potential help in helping public safety authorities that want to locate certain individuals such as wanted criminals, suspected terrorists, and missing children.

The face recognition algorithms can be classified into two major categories: "feature based" and "appearence based". The former includes the algorithms that extract features using facial descriptors like area, angles and distances between characteristic points of the face. In the latter, algorithms based on global properties such as variety and intensity colors of the face images are present.

One of the most widespread algorithm, that belongs to the appearance based category, is the Principal Component Analysis (PCA). It is considered to be one of the first working facial recognition technology, and it served as the basis for one of the top commercial face recognition technology productsIt is said to be a simple, non-parametric method for extracting relevant information from confusing data sets. PCA provides a roadmap for how to reduce a complex data set to a lower dimension to reveal the sometimes hidden, simplified structures that often underlie it. The dimensionality reduction is made identifying the most meaningful basis to re-express a data set. Features are represented as q-dimensional vectors, where each dimension is a sort of measure. PCA projects them into m-dimensional vector with $m < q$, where the m dimension are chosen based on their variances, that is the dimensions where corresponding values of different vectors are more scattered (higher variances). These projections are called “eigenfaces” and here we can find the step-by-step method to compute them (59). For a detailed explanation of the theoretical which the algorithm is based to we suggest (60).

Others important face recognition algorithms used in the literature are the Linear Discriminant Analisys (LDA), Indipendent Component Analisys (ICA), Local Feature Analisys (LFA), along with other algorithms that use Correlation Filters and Neural Networks.

4.2. Fingerprint Recognition

A fingerprint is the pattern of ridges and furrows on the surface of a fingertip. The biological properties of fingerprints are well understood which are summarized as follows: (i) individual epidermal ridges and furrows have different characteristics for different fingerprints; (ii) the configuration types are individually variable, but they vary within limits which allow for systematic classification, (iii) the configurations and minute details of individual ridges and furrows are permanent and do not change with time.

The uniqueness of a fingerprint is exclusively determined by the local ridge characteristics and their relationships. Fingerprint matching generally depends on the comparison of local ridge characteristics and their relationships. A total of one hundred and fifty different local ridges characteristics, called minute details, have been identified. These local ridges characteristics are not evenly distributed. Most of them depend heavily on the impression conditions and quality of fingerprints and are rarely observed in fingerprints. The two most prominent ridge characteristics, called minutiae, are (i) ridge ending and (ii) ridge bifurcation. A ridge ending is defined as the point where a ridge ends abruptly. A ridge bifurcation is defined as the point where a ridge forks or diverges into branch ridges. A fingerprint typically contains about 50 to 100 minutiae. For a given fingerprint, a minutiae can be characterized by its type, its x and y coordinates, and its direction θ.

Fingerprint verification consists of two main stages : (i) minutiae extraction and (ii) minutiae matching.

4.2.1. Minutiae extraction

Minutiae are ridge endings or ridge bifurcations. Generally, if a perfect segmentation can be obtained, then minutia extraction is just a trivial task of extracting singular points in a thinned ridge map. However, in practice, it is not always possible to obtain a perfect ridge map. Some global heuristics need to be used to overcome this limitation.

Orientation field detection

The first step in minutiae extraction is an estimate of the orientation field of the input fingerprint image, which improves the performance of minutia extraction. The Rao's algorithm (61) performs as follow :

- Divide the input fingerprint image into blocks of size $W \times W$
- Compute the gradients G_x and G_y at each pixel in each block
- Estimate the local orientation of each block using the following formula :

$$\theta_0 = \frac{1}{2}\tan^{-1}\left(\frac{\sum_{i=1}^{W}\sum_{j=1}^{W} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{W}\sum_{j=1}^{W}\left(G_x^2(i,j) - G_y^2(i,j)\right)}\right)$$

Where W is the size of the block, and G_x and G_y are the gradient magnitudes in x and y directions, respectively. An example of orientation field estimate is shown in Figure 31.

After the orientation field of an input fingerprint image is estimated, a segmentation algorithm which is based on the local variance of gray level is used to locate the fingerprint region.

Ridges detection

Then a ridge detection phase is performed. The most salient property corresponding to ridges in a fingerprint image is the fact that gray level values on ridges attain their local maxima along the normal directions of local ridges. In (62) they convolve the fingerprint region with two masks, $h_t(x, y; u, v)$ and $h_b(x, y; u, v)$ of size 11 X 7 that are capable of adaptively accentuate the local maximum gray level along the normal direction of the local ridge direction.

Minutiae detection

The minutiae detection is made assuming that if a pixel is on a thinned ridge (eight-connected), then it has a value 1, and 0 otherwise. Let (x, y) denote a pixel on a thinned ridge, and $N_o, N_1, \dots \dots, N_7$ denote its eight neighbors. A pixel (x, y) is a ridge ending if $\sum_{i=0}^{8} N_i = 1$ and a ridge bifurcation if $\sum_{i=0}^{8} N_i > 2$. Spurious minutiae such as spikes and breaks are removed by applying a smoothing procedure described in (62). Then the surviving minutiae are treated as true minutiae, and for each of them, the following parameters are recorded:

- x coordinate
- y coordinate
- orientation
- the associated ridge

4.2.2. Minutia matching

The minutia matching is decomposed into two stages : (i) alignment stage, where transformation such as translation, rotation and scaling between an input pattern and a template are first estimated; the input patterns are then aligned with the template according to the estimated parameters; and (ii) matching stage, where both the input pattern and the template are converted to polygons in polar space and an elastic string matching algorithm is used to macth the resulting polygons.

The following algorithm describes the steps which the minutia matching process is performed. For details of each step refer to (62).

Let $P = ((x_1^P, y_1^P, \theta_1^P)^T, \dots \dots, (x_M^P, y_M^P, \theta_M^P)^T)$ and $Q = \left(\left(x_1^Q, y_1^Q, \theta_1^Q\right)^T, \dots \dots, \left(x_N^Q, y_N^Q, \theta_N^Q\right)^T\right)$ denote the M minutiae in the template and the N minutiae in the input image, respectively.

- Match the ridge associated with each input minutia against the ridge associated with each template minutia and align the two pattern according to the matching result
- Convert the representations of template and input minutiae into polar coordinate representations with respect to the corresponding minutia on which alignment is performed and represent them as two symbolic strings by concatenating each minutia in an increasing order of radial angles:

$$P_p = ((r_1^P, e_1^P, \theta_1^P)^T, \dots \dots, (r_M^P, e_M^P, \theta_M^P)^T)$$

$$Q_p = \left(\left(r_1^Q, e_1^Q, \theta_1^Q\right)^T, \dots \dots, \left(r_N^Q, e_N^Q, \theta_N^Q\right)^T\right)$$

  Where r_*, e_*, and θ_* represent the corresponding radius, radial angle and normalized minutia orientation with respect to the reference minutia respectively.
- Match the resulting strings P_p and Q_p with a modified dynamic-programming algorithm described in (62) to find the edit distance between P_p and Q_p.
- Compute the matching score of the template and input minutiae as the minimum edit distance.

$$M_{pq} = \frac{100N_{pair}}{max\{M, N\}}$$

  Where N_{pair} is the number of minutia pairs which fall within a giveng bounding box.

The maximum and minimum values of the matching score are 100 and 1, respectively.

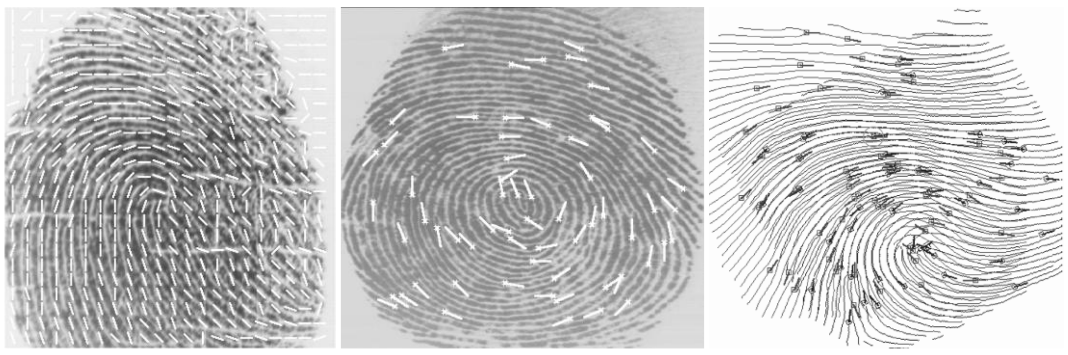

(a) (b) (c)

Figure 31 : (a) Orientation field by Rao's method (b) Minutiae extracted (c) Alignment of input pattern on template pattern

4.3. Iris Authentication

The iris is thought as one of the more reliable and invariant biometric traits, since it does not suffer of significant changes with aging. It has been also widely developed, along with face and fingerprints, in many commercial recognition systems which brands include LG, Oki, Panasonic, Sagem, IrisGuard, Sarnoff, IRIS, Privium, CHILD Project, CanPass, and Clear (RT).

As for the face authentication, also the iris authentication splits in two subsequently phases: the iris location and then the iris recognition.

4.3.1. Iris Detection

The iris detection is not a simple computer vision problem, since its intensity value is close to the pupil's one and it is often obscured by eyelashes and eyelids. However the pupil has a regular shape and a uniform dark tonality, making it easy to detect. Assuming that pupil and iris are concentric, this is a valid starting point for the automatic detection.

Detection of the pupil can be carried out by: removing noise by applying a median blur, thresholding the image to obtain the pupil, performing edge detection to obtain the pupil boundary and then identifying circles.

To obtain the blurred image, a median filter is used as proposed in (63). The overall effect of the median blur is to reduce the noise and pixel intensity complexity of the iris image without perturbing the edge fidelity of the original image. After applying a dynamic threshold to our median blurred image, the location of the pupil is well defined, and edge detection can extract this information to form a single bit depth image which can be searched for circles. The pupil, iris and upper and lower eyelids are then founded using the following operator :

$$\max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{r,x_0,y_0} \frac{I(x,y)}{2\pi r} ds \right|$$

Where $I(x,y)$ is an image containing an eye, The operator searches over the image domain (x,y) for the maximum in the blurred partial derivative with respect to increasing radius *r*, of the normalized contour integral of $I(x,y)$ along a circular arc *ds* of radius *r* and center coordinates (x_0,y_0). The symbol * denotes convolution and $G_\sigma(r)$ is a smoothing function such as a Gaussian of scale σ. The complete operator behaves in effect as a circular edge detector, blurred at a scale set by σ, which searches iteratively for a maximum contour integral derivative with increasing radius at successively finer scales of analysis through the three parameter space of center coordinates and radius (x_0,y_0,r) defining a path of contour integration (64).

Another way to find the outer boundary of the iris and the eyelids is proposed in (63).

4.3.2. Iris Recognition

Each isolated iris pattern is then demodulated to extract its phase information using quadrature 2-D Gaborwavelets. It amounts to a patch-wise phase quantization of the iris pattern, by identifying in which quadrant of the complex plane each resultant phasor lies when a given area of the iris is projected onto complex-valued 2-D Gabor wavelets :

$$h_{\{Re,Im\}} = sgn_{\{Re,Im\}} \int_{\rho} \int_{\phi} I(\rho,\phi)\, e^{-i\omega(\theta_o-\phi)} e^{-(r_0-\rho)^2/\alpha^2} e^{-(\theta_0-\phi)^2/\beta^2} \rho\, d\rho\, d\phi$$

Where $h_{\{Re,Im\}}$ can be regarded as a complex-valued bit whose real and imaginary parts are either 1 or 0 (sgn) depending on the sign of the 2-D integral: $I(\rho,\phi)$ is the raw images in a dimensionless polar coordinate system tha is size- and translation-invariant and which corrects for pupil dilation; α and β are the multiscale 2-D wavelet size parameters, spanning an eoght-fold range from 0.15 to 1.2 mm on the iris; ω is wavelet frequency, spanning three octaves in inverse proportion to β; and (r_0, θ_0) represent the polar coordinates of each region of iris for which the phasor coordinates $h_{\{Re,Im\}}$ are computed (Fig. 32) .

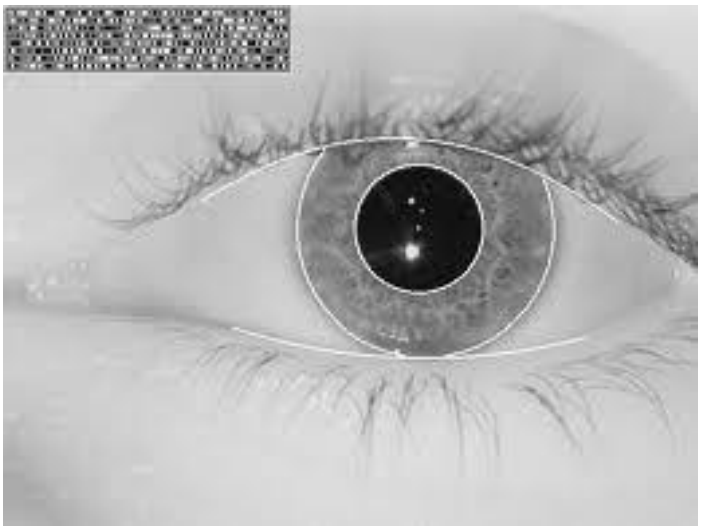


Figure 32 : Iris detection and phase quadrant coding sequence is illustrated by the bit stream shown graphically

Iris recognition is an automated method of biometric identification that uses mathematical pattern-recognition techniques on video images of the irides of an individual's eyes, whose complex random patterns are unique and can be seen from some distance. Almost all the iris recognition uses cameras that acquire images of the iris in the visible wavelength VW (400-700 nm) or near infrared range NIR (700 - 900 nm). The majority of iris recognition systems operate within the longer NIR spectrum which can penetrate dark-coloured irides, the dominant phenotype of the human population, revealing texture not easily observed in the VW spectrum. The NIR spectrum also reduces iris pattern contamination by blocking ambient corneal reflections.

On the other hand, the effects of melanin, the primary colouring component in irises, cannot be distinguished by NIR cameras, that instead are excited by shorter wavelengths within the VW spectrum, providing rich sources of information mainly coded as shape patterns in iris (Fig. 33).

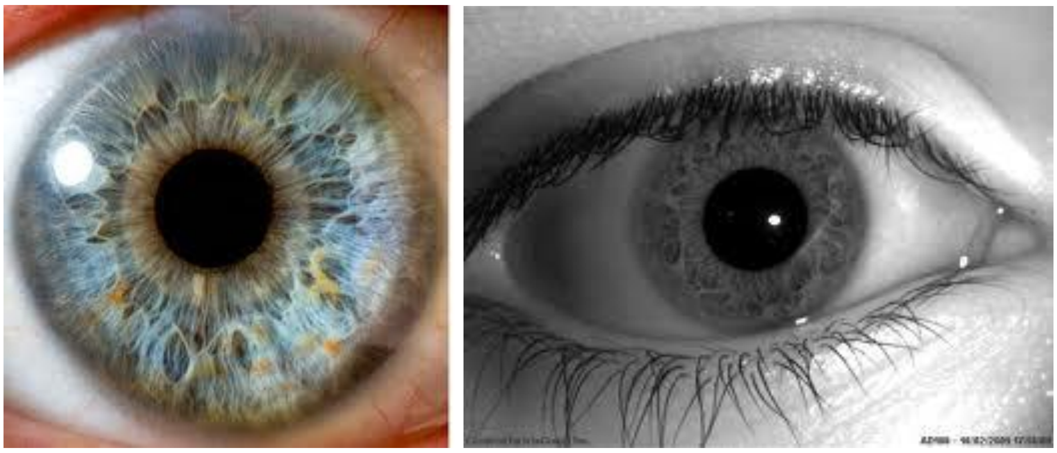


Figure 33 : Two irises captured with VW and NIR cameras respectively.

A key advantage of iris recognition, besides its speed of matching and its extreme resistance to False Matches, is the stability of the iris as an internal, protected, yet externally visible organ of the eye.

5. Software implementation

We are going to talk about how the project has born, which motivations pushed it up, which constraints I encountered, the not successful software tests and an overview of the functions implemented.

5.1. So far Summary

In the previous chapters we talked about unimodal biometrics, about their advantages and disadvantages. The general working scheme has been shown, calling out that one of the major problems that draw attention is how to reach a certain accuracy level in a biometric system authentication. FAR, FRR, EER and ROC curve have been presented, along with intra-class variations and inter-class similarities.

After that the most biometric traits used in the biometric scenery have been presented : DNA, Ear, Iris, Hand geometry, Palmprint, Fingerprint, Face, Vein patterns, and so on (physiological VS behavioral traits), each of them having some advantages and disadvantages.

In order to chose which characteristic to use in the system, some issues have been pointed out and a trade-off among some factors needs to be taken :

- User acceptability
- Security
- Complexity implementation
- Costs
- Legal Issues
- Environment in which the system will work

Multimodal systems then have been analyzed. The use of multiple sources of information has the main advantage to make the system more effective in malicious rejections. But complications have been found compared to the unimodal case. Data need to be combined in a single result, useful to make the final decision about the user validity.
The following levels of fusion have been identified, which the first four are the majors:

- Fusion at sensor level
- Fusion at the feature level
- Fusion at the match score level
- Fusion at the decision level
- Fusion at the rank level

The most common is the fusion at match score level. Densities, classifier and transformation based techniques are the three categories in which all match-score level fusion techniques can be included. But usually densities and transformation methods require a previous normalization phase, in order to put all the data coming from different sources in a common domain. Moreover, densities based systems need a previous accurate match score (genuine and impostor) density estimation to reach good results in accuracy terms.

Many systems have been developed during years, differentiating each other on the chosen techniques of feature extraction, template matching, normalization scores and fusion schemes.

Some well known methods have been proposed. Weighted Sum rule, Generalized densities with Likelihood ratio and Quality sample measures to use as bias of the physiological evidence are often used. It seems that weighted sum rule (transformation) along with SVM rules (classification) are actually the most performance methods in terms of accuracy.

Then three recognition modalities, used in this thesis, have been quickly shown, in order to give a hint on the basis of how face, iris and fingerprint recognitions work.

5.2. Introduction

The main scope of this thesis is to develop an part of a software that will be later used as an complete authentication system based on individual's biometric sources.

The idea is to set it up in order to allow the access in certain areas, to users that have been previously successful enrolled. It should be able to embed new algorithms that at least perform feature extraction and template matching of biometric traits. This because it is believed that new better algorithms will be developed in future, thus it will be possible to substitute old algorithms (or embed them) with new algorithms.

The system should be ready to face with uncommon cases, such as persons with ruined fingerprints or long hairs which hide face portions, or with different eye shape (different races such as Chinese).

In particular the project aims to give the following functions :

- An enrollment function that allows the registration of new users adding them to a database. It also needed an understandable user interface that leads the user while processing his face, iris and fingerprint traits.
- A verification function that allows the recognition of genuine or impostor users. The verification phase has to be as fast as possible to avoid bore user.
- The possibility to insert new algorithms capable of feature extraction and matching templates.
- The possibility to test an algorithm on a database of images (depending of the biometric trait which the algorithm is made for) and retrieve genuine and impostor matching scores graphs as well as FAR and FRR graph.
- Calculate minimum and maximum value scores, and EER of each algorithm when asked
- The possibility of modify user's data.

- The possibility of add a biometric trait .
- The possibility to set user specific thresholds, taking into account that not all the biometric traits have the same degree of availability for any user.

The project can be placed in the searching area called “Identity management”. In our case we used biometrics as tool, since that multiple physiological characteristics are used to store an user’s identity.

5.3. Devices

We used three different devices (sensors) to capture face, fingerprint and iris samples. All of them connected simultaneously to a mobile personal computer.

5.3.1. Iris sensor

The first device is an iris sensor named VistaFA2 produced by Vista Imaging Incorporated (65).

Vista Imaging’s VistaFA2 is a complete multimodal biometric peripheral for Windows CE, XP, Vista, Windows 7. The VistaFA2 uses the highest quality state of the art digital imaging technology to provide a superior biometric imaging solution.
VistaFA2’s iris capturing features include top of the line high resolution CMOS sensor for ISO/IEC 19794-6 compliant images of your iris, custom optics, multi wavelength IR illumination, tri-color RGB LED indicators coupled with their proprietary distance sensing and accurate focus analysis provide unique auto acquisition capabilities. Taken together these features provide an iris imaging solution which result in ease of use and clearer images in nearly all lighting conditions.

Furthermore capturing iris images, the VistaFA2 sensor also allows to use it as a common face camera
VistaFA2’s face capturing features include 5MP CMOS images sensor (2560 x 1920) with special optics for clearer images, providing the highest quality ISO/IEC 19794-5 compliant images everytime.
VistaFA2 provides audio input and optional output for superior audio I/O features.

Figure 34 : An image of VistaFA2 iris/face sensor produced by Vista Imaging Incorporated

5.3.2. Fingerprint sensor

The second sensor is a fingerprint image capture device named FX3000, that allows to capture fingerprint images on a optical surface. Fx3000 is a new smart scanner capable of processing and recognizing fingerprints on board (Match-on-Board). Fx3000 can store internally, in a safe area, templates, files and passwords and is able to perform cryptographic operations. Fx3000 is the ideal solution for digital signature (biometrically enabled) applications: the PC sends to the scanner the document to sign, specifying which certificate is to use; access to the private signature key is granted once the user has been authenticated by his/her fingerprint. Fx3000 is the ideal solution also for Single Sign-On (or “passwords bank”) applications, as it can store internally, in the safe area, the passwords that have to be provided to third party applications once the user has been biometrically authenticated.

A large sensing area improves recognition performance since a larger number of minutiae can be detected and the overlapping area of two fingerprint images, taken in different moments, is significantly maximized. This increases the overall accuracy (smaller FAR) and, above all, greatly reduces the false rejections (FRR) caused by incorrect finger placements. Furthermore, the higher is the resolution, the more details can be located in the fingerprint pattern.
The images captured by Fx3000 have size of 560x400 pixel and 569 dpi quality. For more information refer to (66).

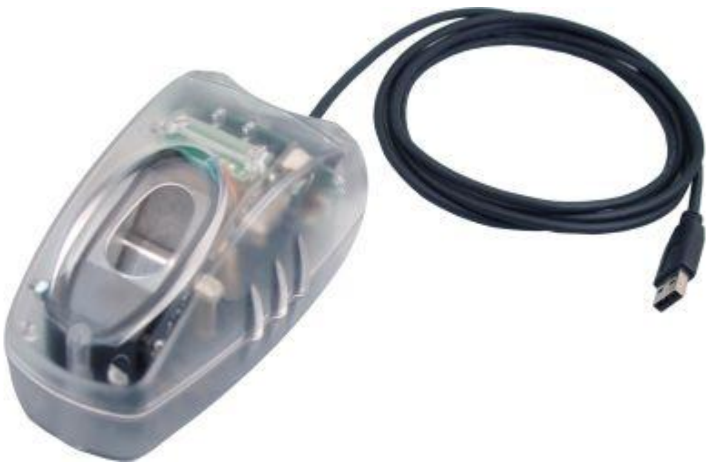


Figure 35 : An image of the FX3000 fingerprint sensor produced by Biometrika

5.3.3. Face sensor

As face image capturing we used the webcam embedded in my HP laptot Pavilion dv6, RAM 4 Gbyte, processor Intel core i7, OS Windows 7.

The camera is a simple webcam named HP TrueVision HD embedded in the laptotp itself. Frames are taken at 1280x720 pixel resolution.

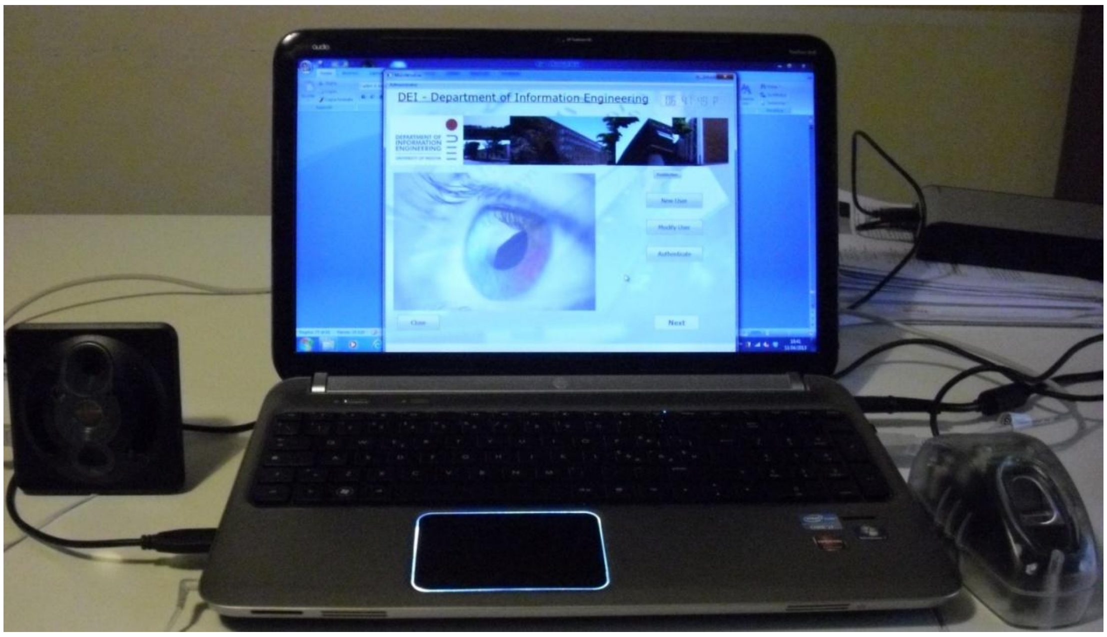


Figure 36 : An image of the workstation used to develop the software

5.4. Implementation

5.4.1. Suit Application choice

Two other works have been made by two colleagues in the field of biometric recognition and software architecture (16) and (67). In their works they used a SDK named Qt Desktop, that uses the famous Qt libraries, mainly known for the graphic interface software. They are a set of multiplatform tools that allow creating applications under Ms Windows, UNIX/Linux, Mac OS X, Embedded Linux, Windows CE/Mobile, Symbian and Maemo.
The SDK uses the C++ language and the C preprocessor, but interfaces for programming languages as Java, Perl, Phyton, C and PHP are available as well. It allows the integration with databases and with XML documents parsing functions.
These were some important reasons for which it has been chosen :

- Possibility of creating simple GUI
- Possibility to develop software using a widespread programming language
- Possibility of interfacing with dababases
- The QT SDK was released under the LGPL license[1]
- Object Oriented Programming (OOP)

Along with Qt SDK, the following softwares have been used

- Intel libraries OpenCV 2.1 or later
- MySQL-5.5.16 or later versions
- Qwt-6.0.1
- Doxygen
- Cmake
- MinGW C/C++ compiler for windows
- Pam-face-authentication (68)
- MegaMatcher Embedded, VeriFinger Embedded, VeriLook Embedded, VeriSpeak Embedded and VeriEye Embedded SDK trial (69)

5.5. First Analisys

First of all we needed to understand how the software would have been operated. The following image (see Fig. 37) shows the initial scheme of what the software would has seemed at the end of the work in an user perspective. First of all we thought it could work in two different manners: white or black box execution. The former is intended to be used by individual with a certain level

[1] At the starting of 2013, Qt SDK has been split into two versions: a commercial one (73) and a free one (74) just with Qt Creator and Qt libraries.

of domain knowledge, for example an administrator that needs to evaluate or to check if the specific algorithm works properly. The latter instead is addressed to people that just need to authenticate and get the access to the system/area he want to enter.

The process starts asking the user his identity, and proceeds in identification mode if no such information is given to the system. After that the choice of the biometric trait is taken and then captured by the properly sensor. Then two ways of authentication can be followed.

If the black-box execution is chosen, then the user just waits for an answer from the system, that can be positive (allowed access) or negative(denied access or another biometric trait is asked). The system will chose feature and matching algorithms by itself and make decision on previously given results (if any). In this case the system will make all the decisions by itself.

If the white-box execution is chosen, the user is asked to make decisions, along the recognition process, about which algorithms to use in each step :

- Choice of the image preprocessing algorithm after an image quality check, in order to enhance the next feature extraction process.
- Choice of the feature extraction algorithm among those available, regarding the biometric trait previously chosen.
- Regardless on the verify or identification case, the matching algorithm is chosen and sample template is compared with the database template selected. In the identification case such operation is made against all the users enrolled.
- Choice of the matching score fusion algorithm.
- If the final result it’s not satisfactory then the user is asked to change the matching algorithm, or the feature extraction algorithm, or the algorithm recognition (the biometric trait).

After this first analysis of how the software would have been, we decided to focus on the black-box execution.

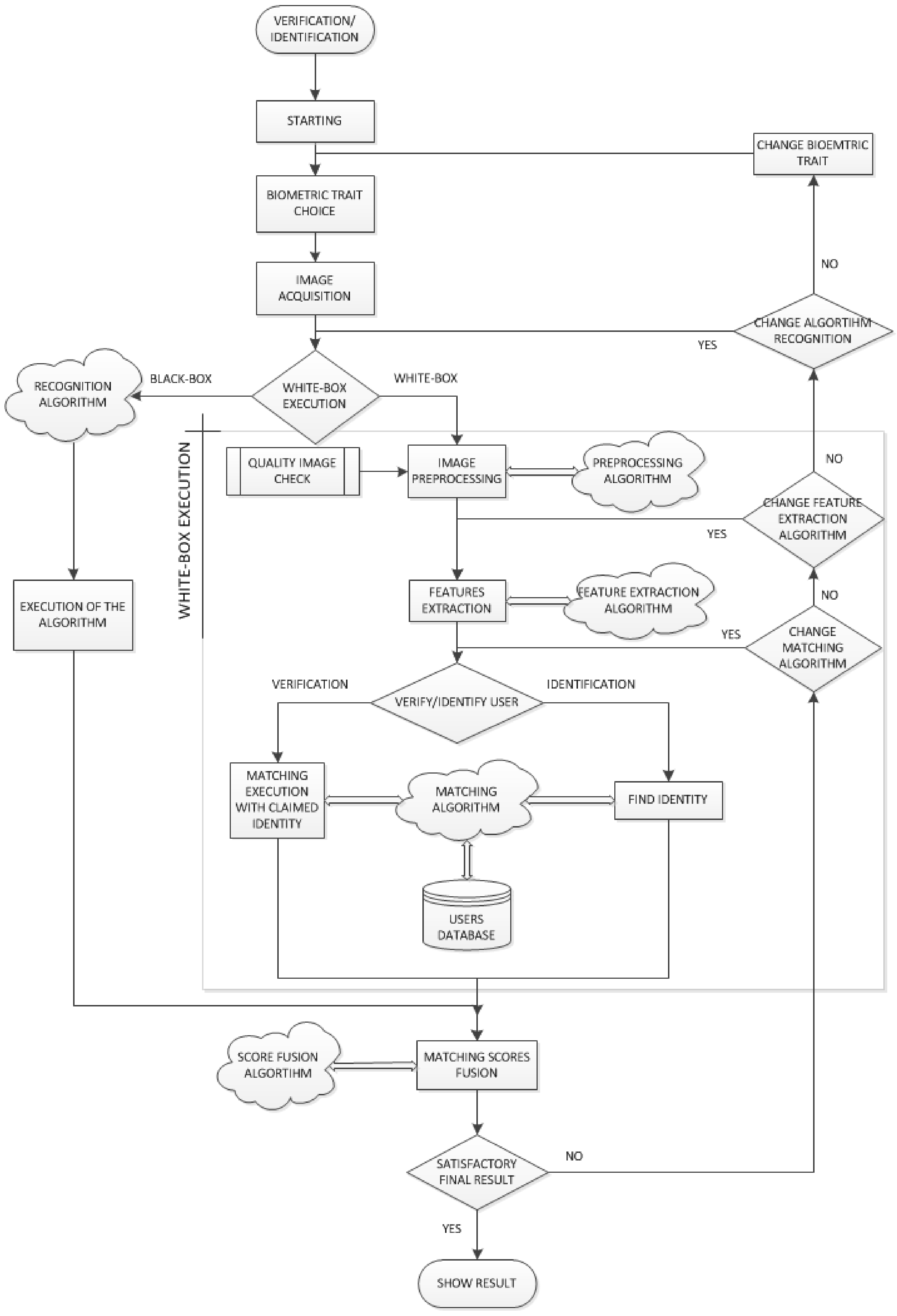


Figure 37 : Representation of how the final system should work

5.6. Black-box execution

The user just has to give the biometric trait requested by the system at that moment, and wait for another recognition phase or for the positive outcome of the previous one. Thus we proposed the functional Enrollment/Verification scheme illustrated in Figure 38.

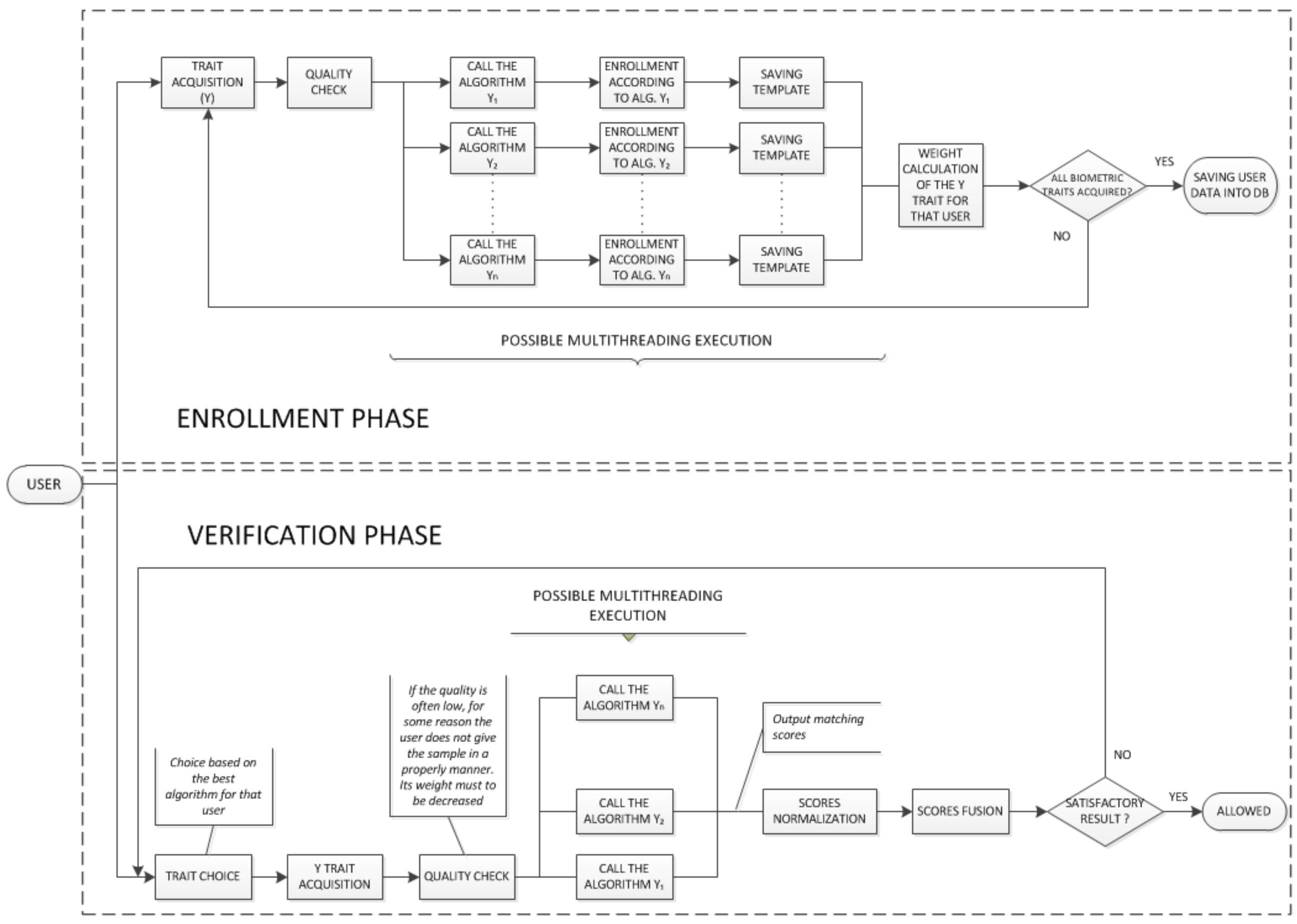

TRAIT ACQUISITION (Y)
QUALITY CHECK
CALL THE ALGORITHM Y₁
ENROLLMENT ACCORDING TO ALG. Y₁
SAVING TEMPLATE
CALL THE ALGORITHM Y₂
ENROLLMENT ACCORDING TO ALG. Y₂
SAVING TEMPLATE
CALL THE ALGORITHM Yₙ
ENROLLMENT ACCORDING TO ALG. Yₙ
SAVING TEMPLATE
WEIGHT CALCULATION OF THE Y TRAIT FOR THAT USER
ALL BIOMETRIC TRAITS ACQUIRED?
YES
NO
SAVING USER DATA INTO DB
POSSIBLE MULTITHREADING EXECUTION
ENROLLMENT PHASE
USER
VERIFICATION PHASE
POSSIBLE MULTITHREADING EXECUTION
If the quality is often low, for some reason the user does not give the sample in a properly manner. Its weight must to be decreased
CALL THE ALGORITHM Yₙ
Output matching scores
Choice based on the best algorithm for that user
CALL THE ALGORITHM Y₂
SCORES NORMALIZATION
SCORES FUSION
SATISFACTORY RESULT ?
NO
YES
ALLOWED
TRAIT CHOICE
Y TRAIT ACQUISITION
QUALITY CHECK
CALL THE ALGORITHM Y₁

Figure 38 : Functional scheme of Enrollment and Verification phases of the system

- Enrollment : in the picture we can see that the user is first asked to enroll one of the biometric traits among those available in the system at that moment (Face, Fingerprint and Iris in our case). Then the biometric source[1] is processed by every available algorithm according to that trait, and a template is saved for each of them. Then a weight to that trait is calculated and the user's data stored in the database.
- Verification : when a user tries to authenticate, the algorithm which has been more successful (greater number of accepted attempts for that user) is selected and put into execution. After the acquisition, an image (source would be more appropriate) quality check is performed to update the weight of that trait. All the algorithms belonging to that trait are called and their output results are normalized and fused giving a final score for

[1] The biometric source is not always an image, since speech recognition require audio files for instance

that trait. Then such a score is fused with other previous results belonging to other traits, if any. If the final score is satisfactory the user is authenticated, if not the next available trait is chosen. If all the biometric traits have been processed and the final outcome was negative, the user is rejected.

5.7. Loading Algorithms

In order to give the possibility of adding algorithms in a later moment, an abstraction of such an algorithm was needed. Generalize the behavior of an object in the OOP environment is not simple.

In (70) they describe simple and elegant solutions to specific problems in object-oriented software design. They give insights that can make software designs more flexible, modular, reusable, and understandable. A lot of generalizing schemes are proposed, and after studying almost all of them we decided that the "Strategy" one was the most suitable for ours problem.

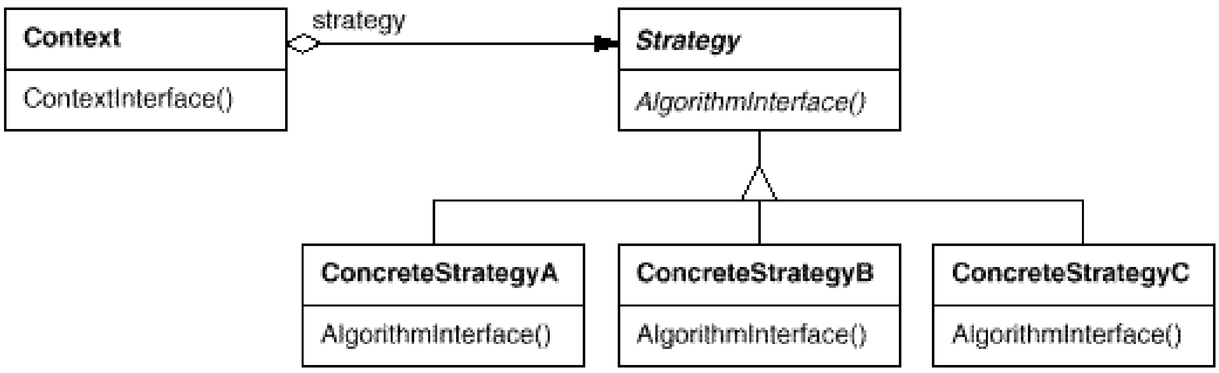


Figure 39 : The Strategy Pattern

They stated that the applicability of such a pattern is when :

- Many related classes differ only in their behavior. Strategies provide a way to configure a class with one of many behaviors.
- You need different variants of an algorithm. For example, you might define algorithms reflecting different space/time trade-offs. Strategies can be used when these variants are implemented as a class hierarchy of algorithms.
- An algorithm uses data that clients shouldn't know about. Use the Strategy pattern to avoid exposing complex, algorithm-specific data structures.
- A class defines many behaviors, and these appear as multiple conditional statements in its operations. Instead of many conditionals, move related conditional branches into their own Strategy class.

The reason for which such a scheme has been chosen is that we needed the possibility to call an algorithm into a context, user verification for example, where some variables were set regarding the context itself (path of user's template and other data) and to use such an algorithm to verify the validity of the user's claimed identity. Since it was not known which type of algorithm will operate at the time of the source code development, a general object algorithm and its method "Verify()" is called. Then depending on the context, at the time of the call, the specific algorithm (iris, face or fingerprint) would perform its implementation of the method with the specific data passed by the context that did the call.

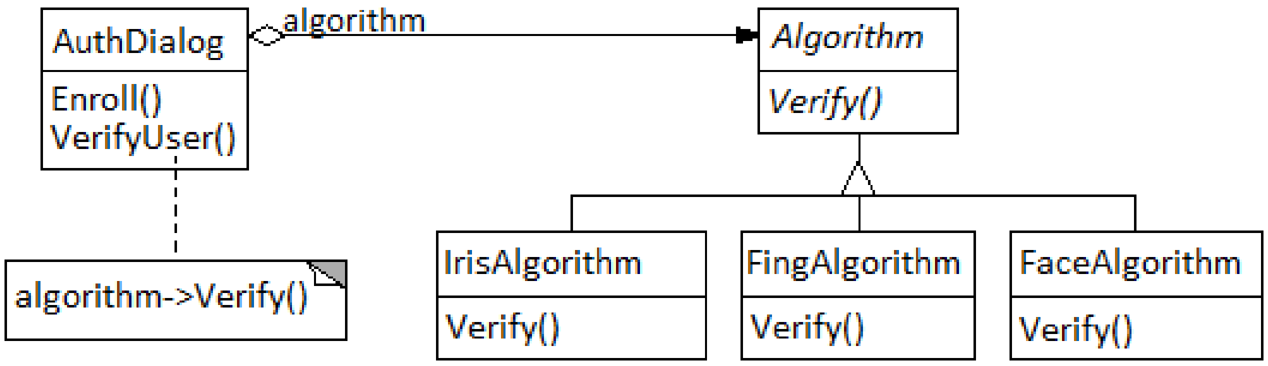


Figure 40 : The Strategy pattern applied to our case of generalizing the biometric algorithm

In our case the context can be a window dialog as part of the GUI, presented to the user. After getting the user identity, the process advances calling the verify method of one of the available algorithms (based on some information such as the best one for that user) and delegates to it the execution of the verification, that will be done according to the type of algorithm chosen.

From shown in Figure 40, the AuthDialog class keeps some instances of the object *Algorithm*. The *Algorithm* interface is subclassed by three objects, each of them having its own "Verify()" (and many others) method implementation. When the user calls the "VerifyUser()" method, pressing a button for example, then the actual instance of *Algorithm* will execute the verification.

As mentioned above, in a later moment we want to add algorithms that compute biometric traits, since there could be improvements in features extraction, matching templates, normalization and fusion score techniques in the future. The tool that gives us such possibility in Qt environment is the QPlugin object along with the DLL libraries :

- QPlugin : it is a container that allows to declare an object as a plugin recognizable by Qt. In this manner the plugin can be loaded at runtime when needed by the software, and subsequently deleted, enhancing the memory allocation.
  During the plugin implementation (say it pluginX), that refers to the implementation of the biometric algorithms in our case, it is asked to :

  - Extends the *Algorithm* interface in pluginX class definition[1] (in pluginX.h file)
  - Use the Q_EXPORT_PLUGIN2() macro in pluginX class implementation (in pluginX.cpp file).
  - Implements all the methods declared in the *Algorithm* interface. This because when the plugin will be loaded at runtime, an instance of it will be created (as shown in the following piece of code) and the compiler requires that all the methods declared as virtual in *Algorithm* interface must to be implemented[2].
  - Put the following lines into the project file[3]
    - TARGET = $$qtLibraryTarget(algorithm_name)
    - TEMPLATE = lib
- DLL libraries : a DLL is a Dynamic-Link Library, that is a software library dinamically loaded at runtime, instead of being statically linked to the execution file during the building. This is exactly what we obtain when compiling the pluginX class with those specific settings as mentioned above. DLL are the libraries running in Windows OS, but if the sources (pluginX) were built in a Linux OS, then the compilation process would create .so files.

Code representing the loading of available algorithms function

```
void MainWindow::loadAlgorithms() //load and instantiate algorithms present in
the algorithms' folder
{
    algorithm *temp;
    QSqlQuery query(db);
    QSqlQuery query2(db);
    QStringList filters;
    #if defined (Q_OS_WIN)
        filters << "*.dll";
    #else
        filters << "*.so";
    #endif
    pluginsDir.setNameFilters(filters);
    pluginsDir.setPath(AlgorithmsPath);
    QString filePath;
        foreach (QString fileName, pluginsDir.entryList(QDir::Files)) {
            filePath = pluginsDir.absoluteFilePath(fileName);
            QPluginLoader loader(filePath);
            QObject *plugin = loader.instance();
            if (plugin)
            { //if the plugin has been instantiated

                AlgorithmInterface *iAlgorithm =
                                       qobject_cast<AlgorithmInterface*>(plugin);
               if (iAlgorithm) //if the plugin implements the AlgorithmInterface
               {
                   algorithms.append(temp);
                   qDebug() << "plugin " << fileName << "loaded" << endl;
                   query.prepare("SELECT * FROM algorithms where path = :path");
                   query.bindValue(":path", filePath);
```

[1] The *Algorithm* class definition requires the Q_DECLARE_INTERFACE() macro
[2] An interface that is extended by subclasses that override its methods, requires to declare them as virtual
[3] For clarifications of how Qt projects work we refer to the Qt Documentation available at http://qt-project.org/doc/

```
                        query.exec();
                        query.next();
                        temp = new algorithm;
                        temp->alg = iAlgorithm;
                        temp->id = query.value(0).toInt();
                        temp->name = query.value(1).toString();
                        temp->path = filePath;
                        temp->sMin = query.value(5).toDouble();
                        temp->sMax = query.value(6).toDouble();
                        temp->EqualErrorRate = query.value(4).toDouble();
                        query2.prepare("SELECT trait FROM traits WHERE id = :id");
                        query2.bindValue(":id", query.value(3).toInt());
                        query2.exec();
                        query2.next();
                        temp->biometricTrait = query2.value(0).toString();
                    }
                }
            }
    synchronizeDatabase();
    algorithmsLoaded = true;
```

Here the files of a predefined folder are selected one by one and checked if they are plugins which implement the *Algorithm* interface (here called "AlgorithmInterface"). If yes, the "iAlgorithm" variable point to a plugin just instantiated that represents a biometric algorithm. The pointer variable is then inserted into a struct that contains algorithm's information, such as EER, min and max score, its path etc., with the aim to completely represents it in the source code environment.

The AlgorithmInterface has been defined as follows:

```
class AlgorithmInterface
{
public:
    virtual ~AlgorithmInterface() {}
    virtual void start() = 0;
    virtual void setTrainingVar(bool ToTrain) = 0;
    virtual bool getTrainingVar() = 0;

    //verify the user giving declared identity, the path of the identity's image
       and the path of the identity's template
    virtual Result verifyUser(string imagePath, string declared_identity,
                              string templatePath) = 0;

    //verify the user giving the query image's path (to extract the template
       from) and the template's path to compare with
    virtual Result verifyUser(string queryImagePath, string templatePath) = 0;

    //test the algorithm with usersImages loaded from a database, along with
       corresponding identities
    virtual tested* test(std::list<string> &usersImages,
                         std::list<string> &usersClasses) = 0;

    //get the genuine and impostor matching scores from usersImages loaded from
       database along with corresponding identities
    virtual void getMatchScoreDistributions(std::list<string> &usersImages,
                                            std::list<string> &usersClasses,
                                     std::vector<double> &genuineMatchingScores,
                                     std::vector<double> &impostorMatchingScores) = 0;

    //identify user just giving his trait's image
    virtual string identifyUser(string imagePath) = 0;

    //train the algorithm in the case of future SVM classification
    virtual void train(list<string> &images, list<string> &classes) = 0;

    //train the algorithm giving a list of general parameters. The aim is to
       generalize as much as possible subjects and classes they belong to
    virtual void train(list<Parameter<class T1, class T2> > &parameters) = 0;

    //create template from sourceImagePath and put it in templateDestinationPath
    virtual void createTemplate(string sourceImagePath,
                                       string templateDestinationPath) = 0;

    //enroll the user with userId, creating template from the image it will
       acquire and saving them in imagePath and templatePath respectively
    virtual bool enroll(string imagePath, int userId, string templatePath) = 0;

    //enroll the user giving the temporary image's path just acquired and the
        template's path where to save the extracted template
    virtual bool enroll(string tempImagePath, string templatePath) = 0;

    //get the time in ms, required to make the verification (from user's verify
       click button to the algorithm answer)
    virtual qint64 getVerifyTime() = 0;

protected:
     bool trainVar;
     MainWindow * parent;
     list<Parameter<class T1, class T2> >* parameters;
};
Q_DECLARE_INTERFACE(AlgorithmInterface,
                          "AlgorithmInterface/1.0")
```

At system startup a synchronization is performed between the predefined folder which contains biometric algorithms, and database which contains information about them (Fig. 42). In fact it would lead the system launching an exception if an algorithm *X* would be present in the folder reserved for algorithms but not in the “Algorithms” table. If *X* is not present in such a folder, than its data are deleted from the database since it is useless keeping them but being not allowed to use the algorithm. Vice versa, if the algorithm is present in the folder but not in the “Algorithms” table than it is automatically added.

Then a GUI has been created to allow user handling algorithms in a easy way. Main functions are :

- Insert a biometric algorithm in the database. When selected from a source folder, the algorithm is automatically copied into the folder reserved for them.
- Delete an algorithm from the database and from its containing folder.
- Test an algorithm on a database and get FAR, FRR and EER accuracy parameters with the threshold at which it reached that value.
- Get a plot of the impostor and genuine distributions of a selected algorithm.

5.8. Database

We started modeling the database for the system by creating the ER-model and switching it to table form. After that we normalized it with the help of the 1NF, 2NF, 3NF and BCNF normal forms. The final result and related tables are described in Figure 41 :

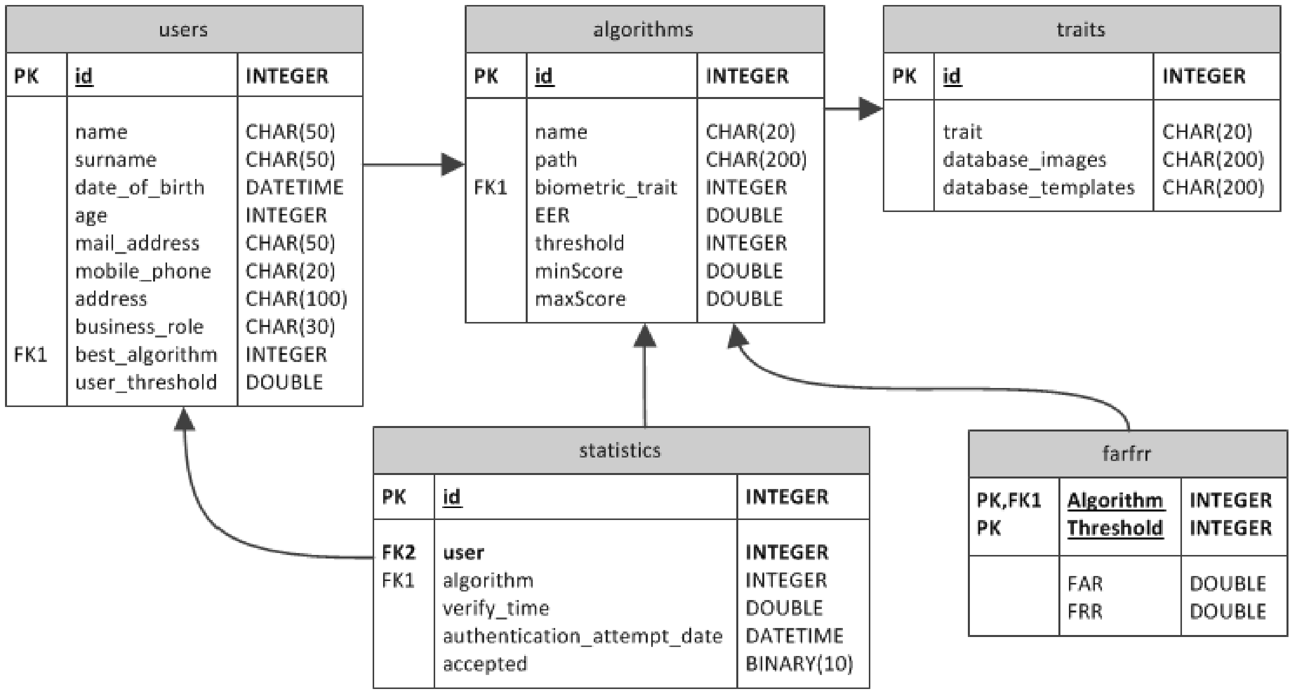
- The users table obviously represents an user and his information. For a user is set a best_algorithm value that refers to the actual best algorithm. The best algorithm of an user is chosen based on the average time that it takes to verify him. This average time is calculated taking into account the average time spent in positive authentications mixed with the number of rejects[1].
- The algorithms table represents a biometric algorithm that implements the *Algorithm* interface. It has name, minScore, maxScore, EER and threshold fields which are obtained during test phase described later. The threshold field refers to that value of threshold obtained to a certain EER value. The biometric trait field refers to the physical characteristic which the algorithm is made for.
- The traits table represents a trait that can be processed and used in the biometric system. Database_images and database_templates refer to the location of enrolled users images and templates respectively.
- Statistics table has been created to store information about user verification attempts. For every attempt date, time, positive or negative result and time taken to verify the user are stored, along with the algorithm used and the user that tried to authenticate.
- The farfrr has been created to store, for each algorithm, the FAR and FRR values at every threshold obtained after testing the algorithm on a test database.

Qt integrates the database connection functions using some self-provided drivers, basically tools interfaces that allow to connect with the underlying databases (for supported database types refer to http://qt-project.org/doc/qt-4.8/sql-driver.html)

The classes that allows to create connections with a database and to execute SQL queries are :

- QSqlDriver : it is an abstract base class for accessing a specific SQL database
- QSqlDatabase : The QSqlDatabase class provides an interface for accessing a database through a connection. An instance of QSqlDatabase represents the connection. The connection provides access to the database via one of the supported database drivers
- QSqlQuery : The QSqlQuery class provides a means of executing and manipulating SQL statements. It can be used to execute DML (data manipulation language) statements, such as SELECT, INSERT, UPDATE and DELETE, as well as DDL (data definition language) statements, such as CREATE TABLE.

[1] An attempt regarding a single trait is labeled as reject if at the end of the verification phase the user is labeled an impostor user.



users		
PK	id	INTEGER
	name	CHAR(50)
	surname	CHAR(50)
	date_of_birth	DATETIME
	age	INTEGER
	mail_address	CHAR(50)
	mobile_phone	CHAR(20)
	address	CHAR(100)
	business_role	CHAR(30)
FK1	best_algorithm	INTEGER
	user_threshold	DOUBLE

algorithms		
PK	id	INTEGER
	name	CHAR(20)
	path	CHAR(200)
FK1	biometric_trait	INTEGER
	EER	DOUBLE
	threshold	INTEGER
	minScore	DOUBLE
	maxScore	DOUBLE

traits		
PK	id	INTEGER
	trait	CHAR(20)
	database_images	CHAR(200)
	database_templates	CHAR(200)

statistics		
PK	id	INTEGER
FK2	user	INTEGER
FK1	algorithm	INTEGER
	verify_time	DOUBLE
	authentication_attempt_date	DATETIME
	accepted	BINARY(10)

farfrr		
PK,FK1	Algorithm	INTEGER
PK	Threshold	INTEGER
	FAR	DOUBLE
	FRR	DOUBLE

Figure 41 : Database tables

The database is composed of two parts :

- The MYSQL part where all the above tables are implemented
- The part represented by application's subfolders as showed by the two schemes in the next section, plus the folder where libraries that implement biometric algorithms are stored

All these tables are checked when the system is launched. If for some reasons they are not present into the database, then they are created (but previous data will be lost anyway). Backup data and security systems could be created in order to secure the system against unpredictable data loss.

5.9. ENROLLMENT

An initial free algorithm has been intensive tested and reorganized, with the aim to make it a plugin corresponding to the interface we presented above. This Face Recognition algorithm can be found at (68) and it is well described in (16). Its recognition engine has been made using the Viola et al. Algorithm, along with a cascade combination of Local Binary Pattern (LBP) algorithm first and Minimum Average Correlation Energy (MACE) algorithm then. Authentication thresholds are variables and dinamically set according to values passed to the algorithms.

OpenCV libraries and theirs trained HAAR LIKE FEATURES for face detection based on the Viola et al. and AdaBoost classifier have been used to first detect face. In OpenCV are available trained classifiers to detect :

- Face without glasses (“haarcascade.xml”)
- Eyes with glasses (“haarcascade_eye_tree_eyeglasses.xml”)
- Eyes without glasses (“haarcascade_eye_.xml”)
- Nose (“haarcascade_nose.xml”)

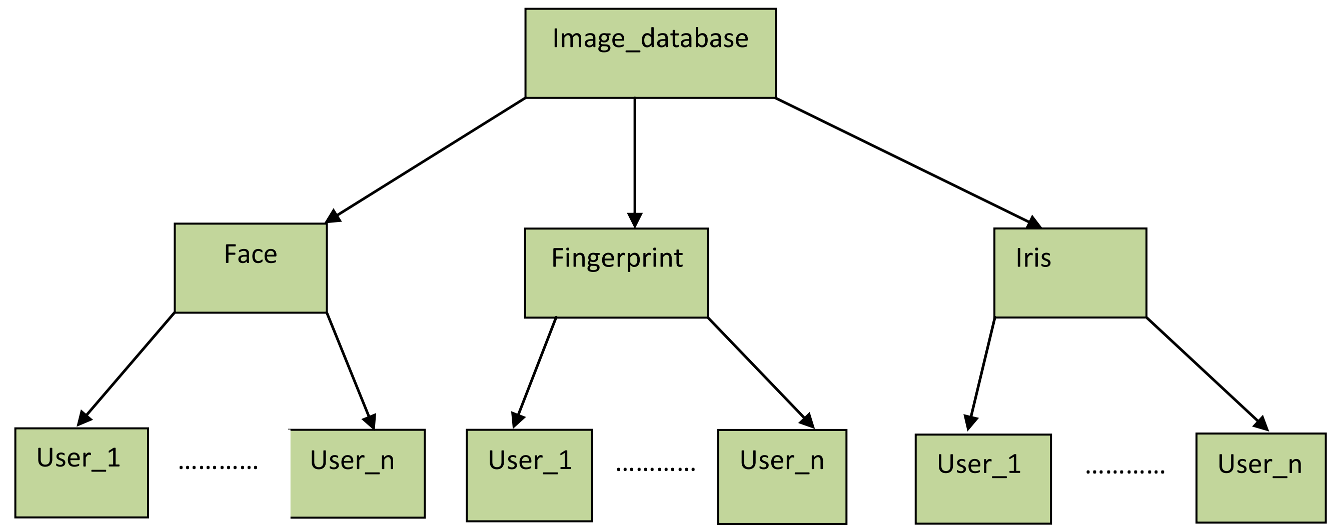
The face authentication algorithm proceeds taking the above objects detected from the image and passing them to the LBP module, which computes the corresponding LBP image and, if in verification phase, calculates the Chi square distance from the stored template, giving greater weights to significant areas. If the resulting value is less than a threshold (the value computed is a dissimilarity value) then the user is authenticated at this stage.

After that a second authentication stage is executed. In Enrollment phase a set of images is extracted from streaming video and for each of them three correlation filters are calculated (face, eyes and nose) which are in turn used to calculate the PSLR of the three regions. Then a unique PSLR threshold for each of the three regions is computed. In Verification mode these three PSLR thresholds will be compared with the values retrieved after performing the same process to a test image coming from an user trying to authenticate. If the value is greater than a percentage set by the administrator then the user will be authorized. (For more details about how correlations filters, PSLR thresholds, LBP image and Chi square distance are calculated, refer to (16)).

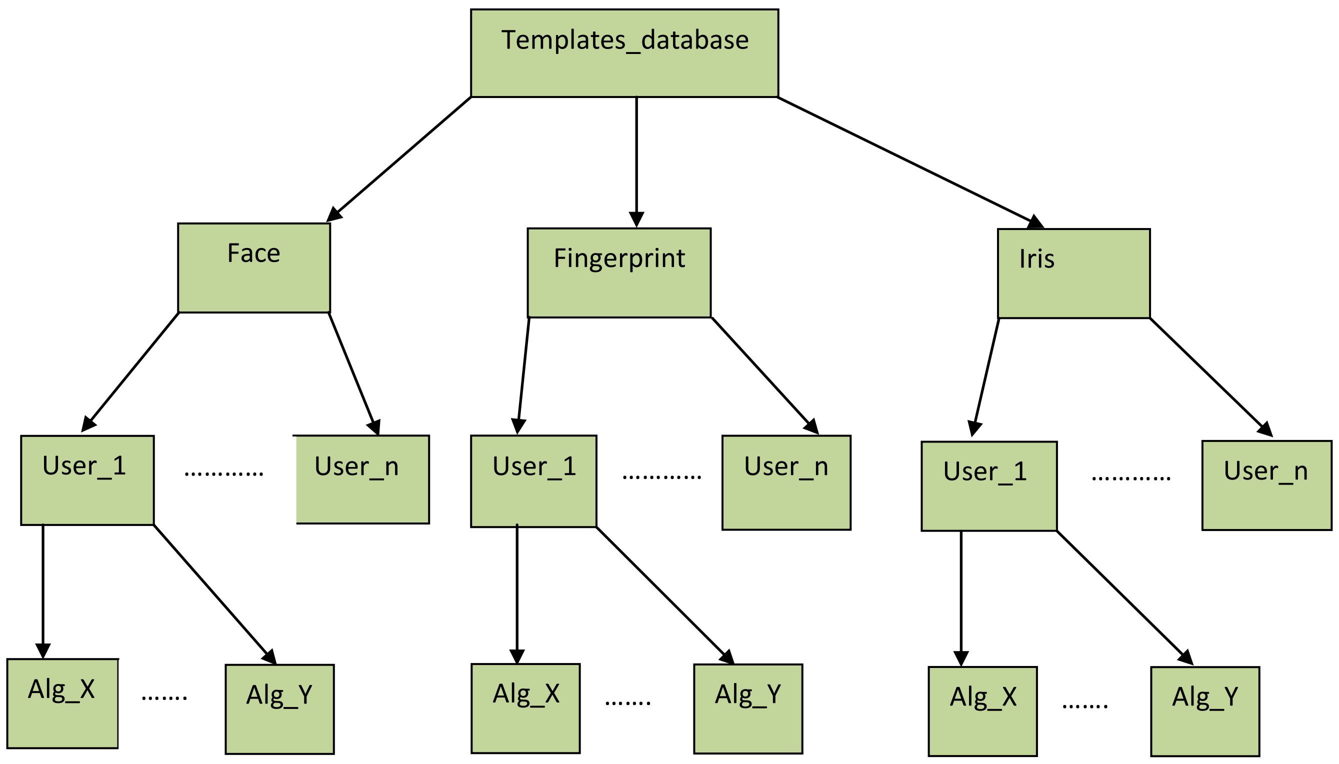
This algorithm had a first main drawback that it was set to compute only live face recognition. There was no databases suitable for how this algorithm enrolls users, thus it was not possible to get some parameters as min-max scores, FAR, FRR and EER. The second was that it suffers, in our tests, of illumination changes. Using the threshold which it came with, an user was not able to be recognized almost all the times if the environment illumination changed in a quite significant way.

After a lot of other tests with other free algorithms, we decided to use libraries licensed by Neurotechnology (69). In Figure 42 the initial screen of the software is showed. Here an individual can decide to try enroll himself because he is a new user, or try to authenticate to the system. He can modify his stores data as well, and an administrator (upper left corner of the screen) can access to the handle algorithms screen.

In the enrollment case, user data are subsequently asked (Fig. 43) and a new user is temporarily created, along with folders that will contain enrollment images and templates. In fact when an user is created, his information are not just stored into the MYSQL database, but images and templates of his traits are collected in apposite folders divided by user and by algorithm that has created them, as shown in the following two schemes. If the enrollment process fails, the temporarily folders will be deleted and the database will not be updated.



Tree scheme folder of images database organization. The one-more level division of algorithms is not requested since images are saved using standard formats and do not depends on the algorithm that processed them, as in case of templates instead



Tree scheme folder of templates database organization.

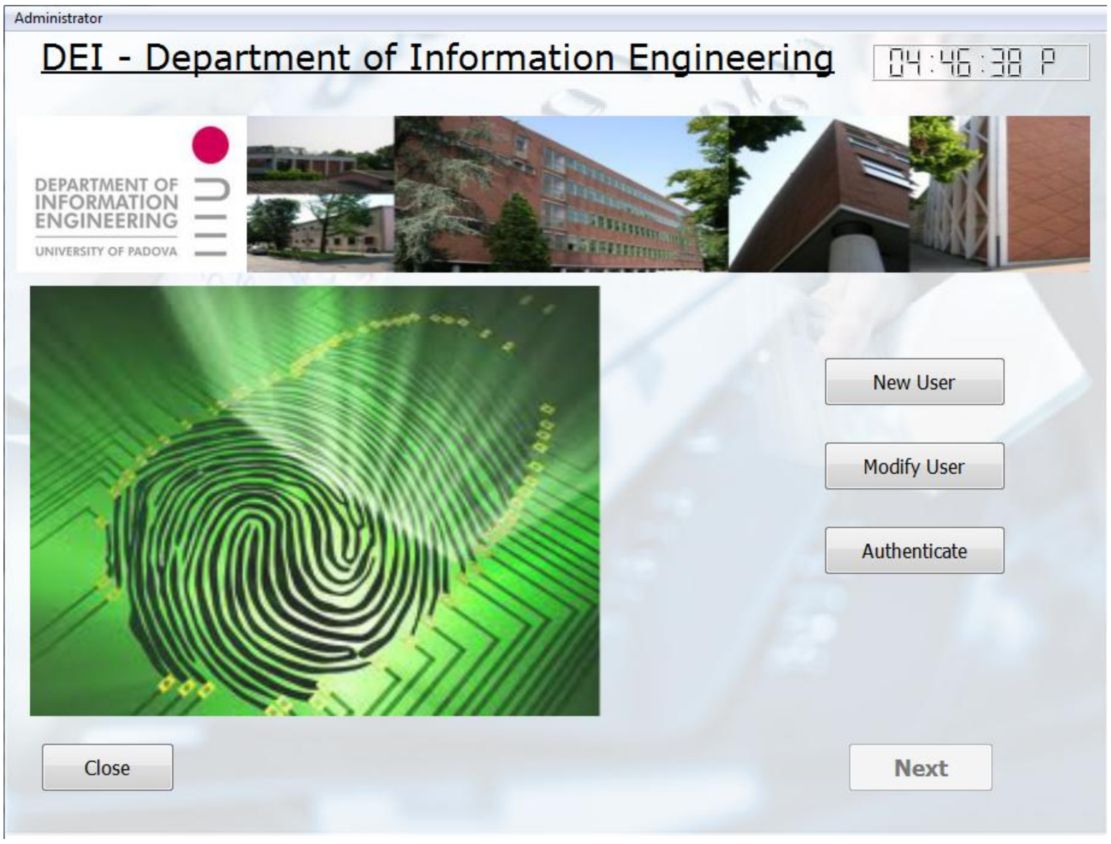
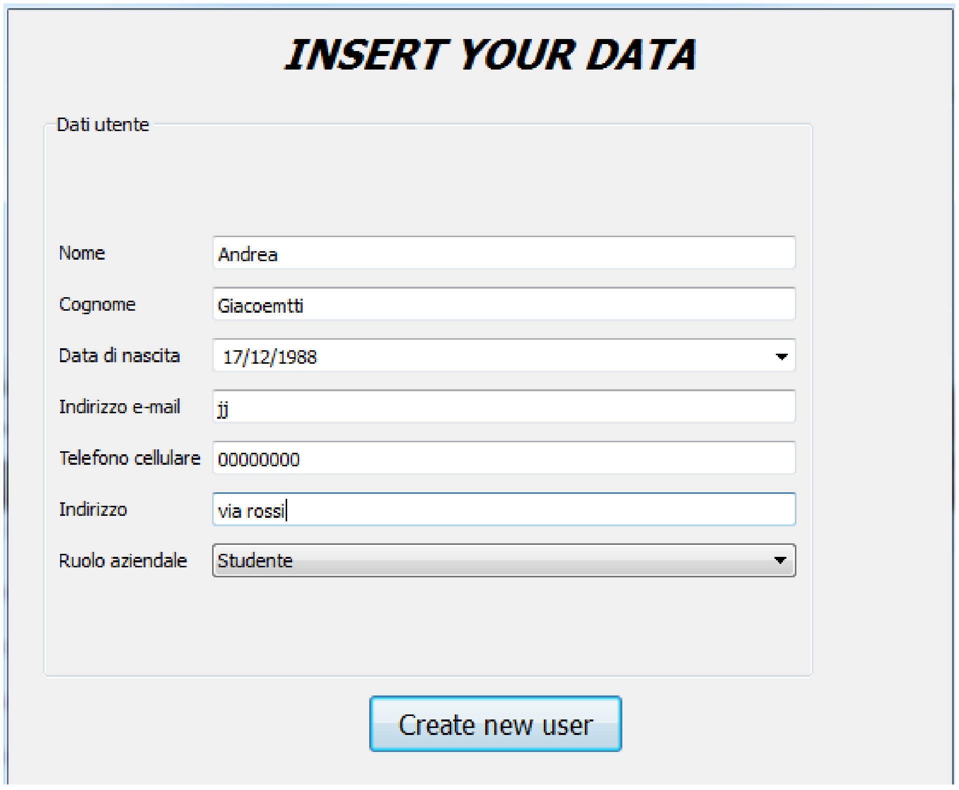


Figure 42 : Software Initial screen


INSERT YOUR DATA

Dati utente

Nome	Andrea
Cognome	Giacoemtti
Data di nascita	17/12/1988
Indirizzo e-mail	jj
Telefono cellulare	00000000
Indirizzo	via rossi
Ruolo aziendale	Studente

Figure 43 : Insert user data screen

5.9.1. Face enrollment

The user enrollment process starts with the face trait. When the user starts the enrollment, the first biometric algorithm in the list of available algorithms (face recognition in our case) is used. Its method “enroll” is called and paths initialized following to the two schemes presented above and according to the user ID and biometric algorithm are passed to it.

Using the OpenCV libraries, the software starts capture images from the camera, converts them in grayscale format and extracts features. Since the Neurotechnologies libraries are under commercial license, no information about how features and templates are extracted have been given to us. Anyway a set of images (we set it to 10 images) from video stream is created and then they are combined to extract features and face template. The capturing phase is controlled by a timer that, when it expires (every 30 mS), it calls a “displayFrame()” method that first captures the frame, then converts it in a image format suitable for Neurotechnology libraries called “HNimage” and pass it to the “extractTemplate()” method.

Code representing the extraction of a face template along with the face and eyes details

```
void FaceAuthentication::extractTemplate()
{
    NInt baseFrameIndex;
    HNGrayscaleImage hGrayscaleImage = NULL;
    NleDetectionDetails details;
    NleExtractionStatus status;
    result = NImageToGrayscale(previewImage, &hGrayscaleImage);
    if(NFailed(result))
        return;
    if (!extractionStarted)
    {
        NBool detected;
        NleFace face;
        result = NleDetectFace(hExtractor, hGrayscaleImage, &detected, &face);
        if (detected)
        {
            extractionStarted = NTrue;
            result = NleExtractStartEx(hExtractor);
        }
    }
    else
    {
        result = NleExtractNextEx2(hExtractor, hGrayscaleImage, &details,
                                              &status);
        if (NFailed(result))
        {
            qDebug() << "extract next failed (result = " << result << ")!" <<
                        endl;
            qDebug() << "status: " << status << "frame index: " << frameIndex <<
                        endl;
        }
        if(activated)
            frames[frameIndex++] = previewImage; //hImage = NULL;
        if (details.FaceAvailable)
        {
            qDebug() << "found face:" << endl;
        qDebug() << "\tlocation = (" << details.Face.Rectangle.X << "," <<
                    details.Face.Rectangle.Y << ")"
                    ", width = " << details.Face.Rectangle.Width << ",
                    height = " << details.Face.Rectangle.Height <<","
                    "confidence = " << details.Face.Confidence << endl;
        face.x = (int)details.Face.Rectangle.X;
        face.y = (int)details.Face.Rectangle.Y;
        face.width = (int)details.Face.Rectangle.Width;
        face.height = (int)details.Face.Rectangle.Height;
        face.confidence = (double)details.Face.Confidence;
    }
    if (details.LeftEyeCenter.Confidence > 0 ||
            details.RightEyeCenter.Confidence > 0)
    {
        qDebug() << "\tfound eyes:" << endl;
        if(details.RightEyeCenter.Confidence > 0)
        {
            qDebug() << "\t\tright: location = (" <<
                            details.RightEyeCenter.X << "," <<
                            details.RightEyeCenter.Y << ")"
                        ", confidence = " <<
                            details.RightEyeCenter.Confidence << endl;

            eyes.right.x = (int)details.RightEyeCenter.X;
            eyes.right.y = (int)details.RightEyeCenter.Y;
            eyes.right.confidence =
                    (double)details.RightEyeCenter.Confidence;
        }
        if(details.LeftEyeCenter.Confidence > 0)
        {
            qDebug() << "\t\tleft: location = (" << details.LeftEyeCenter.X
                           << "," << details.LeftEyeCenter.Y << ")"
                           ", confidence = " <<
                           details.LeftEyeCenter.Confidence << endl;
            eyes.left.x = (int)details.LeftEyeCenter.X;
            eyes.left.y = (int)details.LeftEyeCenter.Y;
            eyes.left.confidence = (double)details.LeftEyeCenter.Confidence;
        }
    }
    if(status != nleesNone && activated)
    {
        result = NleExtractEndEx(hExtractor, &baseFrameIndex, &details,
                                 &status, &hTemplate);
        HNLRecord nlRecord;
        NByte recordQuality;
        NLTemplateGetRecord(hTemplate, baseFrameIndex, &nlRecord);
        NLRecordGetQuality(nlRecord, &recordQuality);
        if (status == nleesTemplateCreated)
        {
            #ifdef freeMode
            cvReleaseCapture(&camera);
            #else
            result = NCaptureDeviceStopCapturing(
                                      (HNCaptureDevice)currentCamera);
            #endif
            qDebug() << "template extracted" << endl;
            if(enrolling)
            {
                qDebug() << "saving image to file " <<
                                      QByteArray("face.bmp") << endl;
                NChar s[faceImage.size()+1];
                for(int i = 0; i < faceImage.size(); i++)
                    s[i] = (NChar)faceImage.at(i);
                    s[faceImage.size()] = '\0';
                    result = NImageSaveToFileEx(frames[baseFrameIndex], s, NULL,
                                                           NULL, 0);
                    if (NFailed(result))
                        qDebug() << "failed to save image to file, error " <<
                                   result << endl;
                    qDebug() << "image saved successfully" << endl;
                    saveTemplate();
                }
                else
                    if(verifying)
                        emit templateReady();
            }
            else
            {
                qDebug() << "extraction failed, status = " << status << endl;
                frameIndex = 0;
                extractionStarted = false;
            }
        }
    }
    NObjectFree(hGrayscaleImage); hGrayscaleImage = NULL;
}
```

Here the set of 10 images are collected before to create the face template. If they not reach an overall quality, the template is not created and the process is repeated with the next 10 images, till a valid set is collected and the template created (or a close button is clicked). Anyway for each frame the face and eyes locations (if detected) are stored. In “displayFrame()” method, every image captured from the webcam is also converted in another format suitable for the Qt libraries, called “QImage”.

Code representing the conversion from OpenCV image (IplImage) to Qt image (QImage)

```
QImage * FaceAuthentication::convertIplImageToQImage(IplImage *ipl)
{
    if (ipl)
    {
        QImage * image = new QImage(ipl->width, ipl->height,
                                      QImage::Format_RGB32);
        int n,m;
        for (n=0;n<ipl->height;n++)
        {
            uchar * scanLine = image->scanLine(n);
            for (m= 0;m<ipl->width;m++)
            {
                CvScalar s;
                s=cvGet2D(ipl,n,m);
                QRgb value;
                value = qRgb((uchar)s.val[2], (uchar)s.val[1], (uchar)s.val[0]);
               ((uint *)scanLine)[m] = value;
            }
        }
        return image;
    }
}
```

This because the graphic object we use to display images requires such an image object. Then the information about face and eyes locations are used to draw boundaries on the “Qimage” just computed that is being displayed, in order to give a feedback to the user that is using the software (Fig. 44). If the face enrollment is not successful, than a error message is displayed and the process terminated.

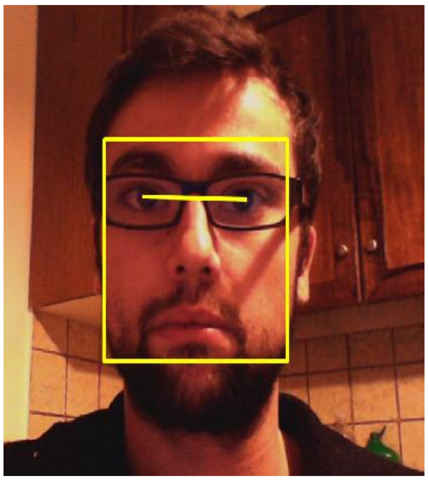
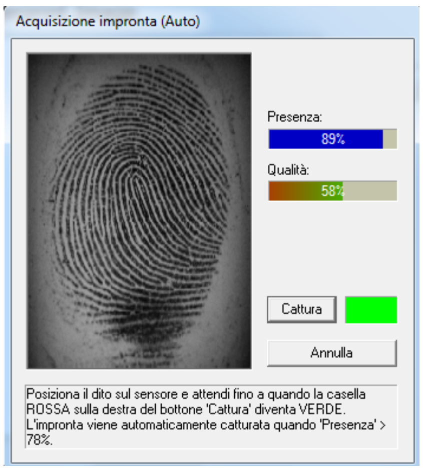


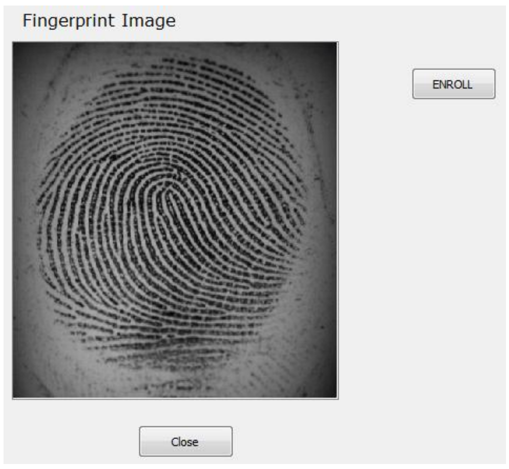
Figure 44 : Boundaries of detected face and eyes locations

5.9.2. Fingerprint enrollment

If the face enrollment has been successful, the actual biometric algorithm is substituted with the next one in the available algorithms list (fingerprint recognition in our case), and its trait enrollment starts. As for the face enrollment, the image captured during enrollment and the template extracted are saved in folders according to the user ID and biometric algorithm. The GUI presents a live capture streaming video during the finger pressure on the sensor surface (Fig. 45 (a)). If the image acquired and saved compares as in Figure 45 (b) then the template has been correctly saved.



Posiziona il dito sul sensore e attendi fino a quando la casella ROSSA sulla destra del bottone 'Cattura' diventa VERDE. L'impronta viene automaticamente catturata quando 'Presenza' > 78%.



(a) (b)

Figure 45 : (a) Live capture video while putting finger on the sensor surface and (b) fingerprint image acquired and saved

5.9.3. Iris enrollment

As the last one, the iris trait is asked to the user if the previous fingerprint enrollment has been successful.

OpenCV was not able to communicate with the VistaFA2 camera, so we had to use native libraries given to us by Vista Imaging. The capturing is divided into two steps :

- In the first step a streaming video of preview images at 640x480 resolution in grayscale format are retrieved using a timer as in face enrollment that every 30 mS calls a “displayFrame()” method to show the preview acquired. The preview acquisition continues till the camera changes a “ready” flag. This happens when some quality measures are held at least for an exposure time interval (set to 400mS). A running thread checks if the ready state is set, and when this happens the second step is performed.
- Some camera setting are changed (“still” mode) and the image that satisfy quality measures according to ready state are captured (at 1280x760 resolution) and the iris image is detected. This is then passed to the features extraction and template creating module where pupil and iris boundaries (Fig. 46) are computed and showed in the final image presented on the GUI as visual feedback for the user

Code representing the initialization of the VistaFA2 camera settings

STEP 1:

```
void DisplayWorker::start()
{

//INIT CAMERA
    // try to open the driver
    rc = VcOpen();
    // turn camera power on
    rc = VcSetControlValue( VC_CTRL_POWER, VC_CTRL_POWER_ON, sizeof(DWORD) );

    //turn the capture off to set params
    rc = VcSetControlValue( VC_CTRL_CAPTURE, VC_CTRL_CAPTURE_OFF, sizeof(DWORD)
);
    //set the video source (face or iris)
    rc = VcSetControlValue( VC_CTRL_VIDEO_SOURCE, VC_CTRL_VIDEO_SOURCE_IRIS,
sizeof(DWORD));
    //set the Automatic Gain Control, the shutter and IR illumination will be
automatically handled
    rc = VcSetControlValue( VC_CTRL_AGC, VC_CTRL_AGC_AVERAGE, sizeof(DWORD));

    rc = VcSetControlValue( VC_CTRL_BACKLIGHT, backlight, sizeof(backlight) );

    rc = VcSetControlValue( VC_CTRL_BLUE_BALANCE, blueBalance,
sizeof(blueBalance) );
    rc = VcSetControlValue( VC_CTRL_BRIGHTNESS, brightness, sizeof(brightness)
);
    //compression images quality is set to 75 (0-100) by default
    rc = VcSetControlValue( VC_CTRL_COMPRESSION_QUALITY, compressionQuality,
sizeof(compressionQuality) );
```

```
    rc = VcSetControlValue( VC_CTRL_CONTRAST, contrast, sizeof(contrast) );
    //camera refresh rate
    rc = VcSetControlValue( VC_CTRL_FLICKER_FREQUENCY,
VC_CTRL_FLICKER_FREQUENCY_60HZ, sizeof(DWORD));
    rc = VcSetControlValue( VC_CTRL_GAMMA, gamma, sizeof(gamma) );
    rc = VcSetControlValue( VC_CTRL_INVERT, VC_CTRL_INVERT_ON, sizeof(DWORD));
    rc = VcSetControlValue( VC_CTRL_MIRROR, mirror, sizeof(mirror) );
    rc = VcSetControlValue( VC_CTRL_RED_BALANCE, redBalance, sizeof(redBalance)
);
    rc = VcSetControlValue( VC_CTRL_SATURATION, saturation, sizeof(saturation)
);
    rc = VcSetControlValue( VC_CTRL_SHARPNESS, sharpness, sizeof(sharpness) );
    //the cameera will attempt to normalize the color gain to achieve a pure
white for bright areas of the camera
    rc = VcSetControlValue( VC_CTRL_WHITE_BALANCE, VC_CTRL_WHITE_BALANCE_AUTO,
sizeof(DWORD));
    //set the audio mode when an iris image is captured
    rc = VcSetControlValue( VC_CTRL_AUDIO_OUT_MODE, VC_CTRL_AUDIO_OUT_MODE_AUTO,
sizeof(DWORD));
    rc = VcSetControlValue( VC_CTRL_AUDIO_OUT_VOLUME, audioOutVolume,
sizeof(audioOutVolume) );
    //set the auto acquire mode on
    rc = VcSetControlValue( VC_CTRL_AUTO_ACQUIRE, VC_CTRL_AUTO_ACQUIRE_ON,
sizeof(DWORD));

    //set the minimum time needed to wait before capturing image when the target
    //is at proper distance from the camera and in focus
    rc = VcSetControlValue( VC_CTRL_AUTO_ACQUIRE_HOLD_TIME, autoAcquireHoldTime,
sizeof(autoAcquireHoldTime));
    rc = VcSetControlValue( VC_CTRL_FLASH_MODE, VC_CTRL_FLASH_MODE_AUTO,
sizeof(DWORD) );
    rc = VcSetControlValue( VC_CTRL_FLASH_ILLUMINATION, flashIllumination,
sizeof(flashIllumination) );
    rc = VcSetControlValue( VC_CTRL_RESOLUTION, VC_CTRL_RESOLUTION_640x480,
sizeof(DWORD));
    rc = VcSetControlValue( VC_CTRL_ZOOM, zoom, sizeof(zoom));
    rc = VcSetControlValue( VC_CTRL_AOI_X_OFFSET, previewAoiX,
sizeof(previewAoiX));
    rc = VcSetControlValue( VC_CTRL_AOI_Y_OFFSET, previewAoiY,
sizeof(previewAoiY));
    rc = VcSetControlValue( VC_CTRL_FORMAT, VC_CTRL_FORMAT_RGB24,
sizeof(DWORD));
    //compressed transfer is not allowed by default
    rc = VcSetControlValue( VC_CTRL_COMPRESSED_TX, VC_CTRL_COMPRESSED_TX_OFF,
sizeof(DWORD));
    //set the possibility to display the iris/pupil location graphics in the
image data
    rc = VcSetControlValue( VC_CTRL_OVERLAY, VC_CTRL_OVERLAY_ON, sizeof(DWORD));
    //set the colors of the LED that indicate the distance of the subbject (blue
= far, red = close, green = ok)
    rc = VcSetControlValue( VC_CTRL_RGB_LED, VC_CTRL_RGB_LED_AUTO,
sizeof(DWORD));
    rc = VcSetControlValue( VC_CTRL_CAPTURE, VC_CTRL_CAPTURE_PREVIEW,
sizeof(DWORD) );

    displayTimer->start(30);
    connect(displayTimer, SIGNAL(timeout()), this, SLOT(displayFrame()));
    readyTimer->start(30);
    connect(readyTimer, SIGNAL(timeout()), this, SLOT(checkReadyStatus()));

}
```

Code representing the capture and location of the iris image

STEP 2 :

```
void DisplayWorker::checkReadyStatus()
{
    DWORD p;
    int rc;
    rc = VcGetControlValue( VC_CTRL_AUTO_ACQUIRE_READY_STATUS, &p, sizeof(p),
NULL );
    if( p == VC_CTRL_AUTO_ACQUIRE_READY)
    {
        //set STILL params
        DWORD zoom = 1;
        DWORD stillAoiX = 0;
        DWORD stillAoiY = 0;
        rc = VcSetControlValue( VC_CTRL_CAPTURE, VC_CTRL_CAPTURE_OFF,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_RESOLUTION, VC_CTRL_RESOLUTION_1280x960,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_FORMAT, VC_CTRL_FORMAT_RGB8,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_ZOOM, zoom, sizeof(zoom));
        rc = VcSetControlValue( VC_CTRL_AOI_X_OFFSET, stillAoiX,
sizeof(stillAoiX));
        rc = VcSetControlValue( VC_CTRL_AOI_Y_OFFSET, stillAoiY,
sizeof(stillAoiY));
        rc = VcSetControlValue( VC_CTRL_OVERLAY, VC_CTRL_OVERLAY_OFF,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_AUTO_ACQUIRE, VC_CTRL_AUTO_ACQUIRE_OFF,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_AUDIO_OUT_MODE,
VC_CTRL_AUDIO_OUT_MODE_MANUAL, sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_COMPRESSED_TX, VC_CTRL_COMPRESSED_TX_ON,
sizeof(DWORD));
        rc = VcSetControlValue( VC_CTRL_CAPTURE, VC_CTRL_CAPTURE_STILL,
sizeof(DWORD));
        //get the image
        PUCHAR pStillImageData;
        int width = 1280;
        int height = 960;
        int nChannels = 1;
        DWORD stillImageSize = width * height * nChannels;
        DWORD retLen;
        pStillImageData = (PUCHAR) malloc(stillImageSize);
        rc = VcGetImage( pStillImageData, stillImageSize, &retLen );
        if( rc == VC_SUCCESS)
        {
            DWORD irisVgaImageSize = SINGLE_IRIS_IMAGE_WIDTH *
SINGLE_IRIS_IMAGE_HEIGHT;
            PUCHAR pIrisVgaImageData = (PUCHAR) malloc(irisVgaImageSize);
            GET_IRIS_IMAGE_PARAMS     irisParams;
            //create STILL BitMap Info
            bmiStill.bmiHeader.biSize = sizeof( BITMAPINFOHEADER );
            bmiStill.bmiHeader.biWidth = 1280;
            bmiStill.bmiHeader.biHeight = 960;
            bmiStill.bmiHeader.biPlanes = 1;
            bmiStill.bmiHeader.biSizeImage = 0;
            bmiStill.bmiHeader.biXPelsPerMeter = 0;
            bmiStill.bmiHeader.biYPelsPerMeter = 0;
            bmiStill.bmiHeader.biClrUsed = 0;
            bmiStill.bmiHeader.biClrImportant = 0;
            bmiStill.bmiHeader.biBitCount = 8;
            bmiStill.bmiHeader.biCompression = BI_RGB;
            // initialize the black and white palette
            for ( int i=0; i < 256; i++ ) {
                 bmiStill.bwPalette[i] = i | ( i << 8 ) | ( i << 16 );
            }
            // fill in the input structure
            irisParams.cbSize = sizeof( irisParams );
            irisParams.pCamImage = pStillImageData; //immagine su cui si cerca e
si taglia l'iride
            irisParams.camImageWidth = bmiStill.bmiHeader.biWidth;
            irisParams.camImageHeight = bmiStill.bmiHeader.biHeight;
            irisParams.irisImageWidth = SINGLE_IRIS_IMAGE_WIDTH;
            irisParams.irisImageHeight = SINGLE_IRIS_IMAGE_HEIGHT;
            irisParams.pIrisImage = pIrisVgaImageData; //immagine che conterrà
l'iride
            rc = VcGetIrisImage( &irisParams );
            if (( rc == VC_SUCCESS ) && ( irisParams.irisInfo.status ==
VC_SUCCESS ))
            {
                bmiIrisVga.bmiHeader.biSize = sizeof( BITMAPINFOHEADER );
                bmiIrisVga.bmiHeader.biWidth = SINGLE_IRIS_IMAGE_WIDTH;
                bmiIrisVga.bmiHeader.biHeight = SINGLE_IRIS_IMAGE_HEIGHT;
                bmiIrisVga.bmiHeader.biPlanes = 1;
                bmiIrisVga.bmiHeader.biSizeImage = 0;
                bmiIrisVga.bmiHeader.biXPelsPerMeter = 0;
                bmiIrisVga.bmiHeader.biYPelsPerMeter = 0;
                bmiIrisVga.bmiHeader.biClrUsed = 0;
                bmiIrisVga.bmiHeader.biClrImportant = 0;
                bmiIrisVga.bmiHeader.biBitCount = 8;
                bmiIrisVga.bmiHeader.biCompression = BI_RGB;
                for ( int i=0; i < 256; i++ )
                {
                    bmiIrisVga.bwPalette[i] = i | ( i << 8 ) | ( i << 16 );
                }
                //rc = VcSaveBmp(fingerprintImage, pStillImageData,
stillImageSize, &bmiStill, sizeof(bmiStill));
                //rc = VcSaveBmp(fingerprintImage, pIrisVgaImageData,
irisVgaImageSize, &bmiIrisVga, sizeof(bmiIrisVga) );
                iris = pStillImageData;
                stop();
            }
        }
    }
}
```

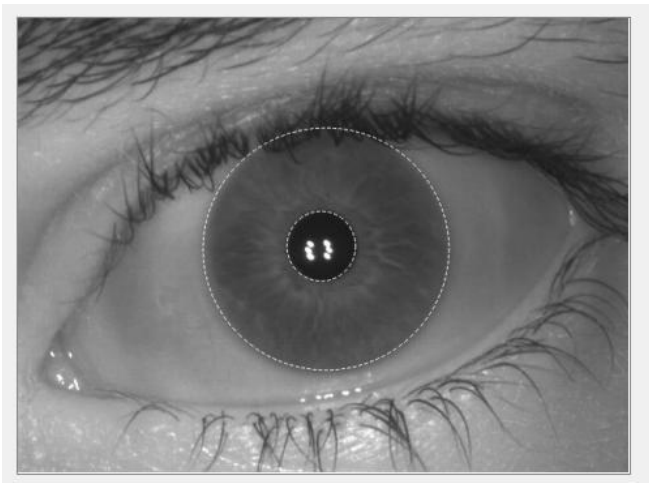


Figure 46 : Boundaries of the detected pupil and iris

5.9.4. Adjust thresholds

It is possible that, among the biometric traits used by the system, an user does not possess one of them (or even more) for some reasons. It is also possible that the enrollment phase was not executed in a good way, and a bad template has been stored (even if in our software every biometric sample captured by one of the three devices needs to reach a certain threshold of quality before acquired).
When this happens, then in a verification scenario, a match score coming from the matcher that is related to the bad user's trait should be made less important of the others coming from matchers which have not noticed such a problem.

We have then put an adjusting user threshold phase after the enrollment one, where the user is asked to make some verify attempts so that single modality thresholds (one for each trait) can be adjusted, along with the final threshold which the fused score is compared to in the verification process.

Every verification attempt is made of three different single verifications (face, fingerprint and iris). Let $A_{i,j}$ be the i-th attempt of the modality j, with $i = 1, 2, \dots \dots, N$ where N is the number of total attempts and $j = 1, 2, 3$ represents the face, fingerprint and iris modality respectively. Then a quality measure $q_{i,j}$ of the template created from the sample just acquired, is calculated and used to infer if the user shows problems while giving that biometric trait. For example if a low quality measure regarding the fingerprint is retrieved in almost all attempts, then the user weight for that modality should be low.

At the same time if the user has been rejected at almost all attempts because the fused score did not overcome the final threshold , this should be decreased and the attempts repeated again.

5.10. Verification

Since biometrics are more and more widely used in recent years in access control scenarios, the verification (or authorization, authentication) function of the biometric system is critical. We recall that verification is in general less complicate rather than identification for reasons presented in section 2.2. Achieving the same verification performances in identification mode is more challenging. Identification is more used into government or police systems to realize the so called "negative recognition", that is find the identity of an individual given his biometric sources, without the individual's claimed one.

Verification instead is more suitable in control access scenarios where a claimed identity by the user comes along his biometric sources.

We decided to face the problem basing on a dynamical sequential fusion architecture. The aim of the proposed architecture is to reduce the invasive degree that the user feel when he uses it, and to reduce the overall average time necessary to authenticate. This is achieved using the least number of systems (biometric traits) as possible. The architecture is sequential because biometric algorithms are used one after another, and dynamical because the number of algorithms used depends on each access.

Such an approach requires the fusion of the output information coming from each algorithm that performed a verification step. We decided to focus on score-level fusion, since at this level we have a good trade-off between development complexity and information richness as said in section 3.3.3.

The starting idea was to add a possible outcome to the "2-classes" decision (genuine-impostor) that is accepting or rejecting a user identity thanks to a unique threshold on the scores of the system. Thus an algorithm would have three possible decisions: acceptance, rejection or no-decision. As suggested in (71) the "no-decision" refers to the case when a score coming from a matching process, lies in the interval between two different thresholds, a low threshold and an high threshold, given by the genuine and impostor matching score densities, as shown in Figure 47.

The thresholds and thus the 2 quasi-certain areas $]-\infty;\ t_l]$ and $[t_h;\ +\infty[$ (t_l stays for low threshold and t_h stays for high threshold) are determined so that only one class (client, impostor) is present. Those high and low scores allow to take a decision by using only the considered algorithm (that is without performing fusion) when a score belongs to the two previously mentioned intervals.

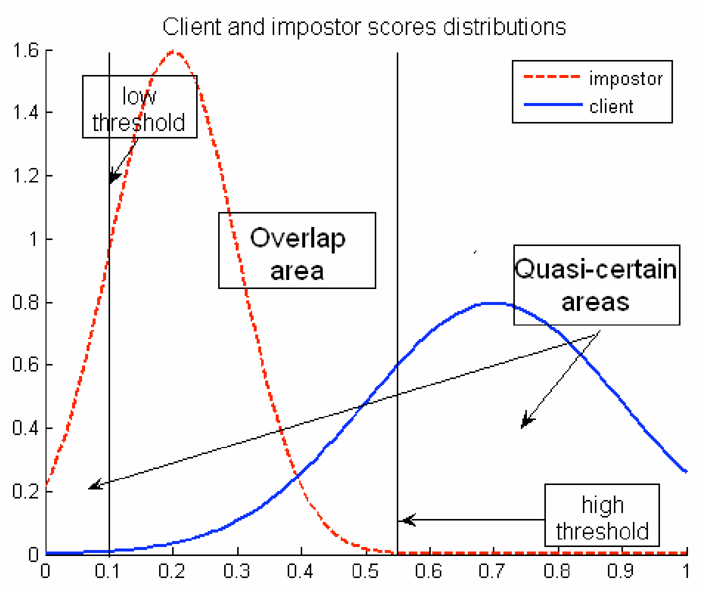


Figure 47 : Three decision areas bounded by two thresholds, based on the overlapping area between genuine and impostor densities

The densities shown in Figure 47 are ideal densities of genuine and impostor scores. In our system no one biometric modality has determined such an overlap area between the two densities (see Fig. 54) , due to the fact that the libraries used to perform features extraction and matching templates are under commercial license and then of high performance. We then proceeded in the common way using a unique threshold.

When an user verify is performed, the user is asked to insert his e-mail address to claim the identity. After that the following steps are executed :

- The best algorithm suitable to that user, say it Alg_i, is selected to start the process. The decision of which has to be used among those available is made basing on previous statistics stored after each authentication attempt made by that user. In fact at the end of every verification, the following data are stored :
    - The algorithm that performed the authentication
    - The user claimed identity
    - The time needed to get an output
    - Date and time of verification attempt
    - The positive or negative result of the verification

  For each algorithm Alg_i, is calculated an average verification time needed by the user that uses it. This average time, say it $Avg_{i,j}$ where i stays for the i-th algorithm and j for the j-th user, changes dynamically since it is updated after each attempt of authentication. The average time is calculated as follows :

$$Avg_{i,j} = \frac{\sum_{s=1}^{K} time_s^{i,j}}{K} + Nrejected_{i,j} \times \max_s\{time_s^{i,j}\}$$

  Where $time_s^{i,j}$ is the time taken by algorithm i to output a positive result (genuine result), when user j tried to authenticate at the s-th attempt, K is the total number of positive attempts made by the user j using the algorithm i and $Nrejected_{i,j}$ is the number of times the user j has been rejected by the algorithm i. The time verification is the time interval starting at the moment the user click on a “verify” button, till an output decision is taken. If the user is accessing the system for the first time, then the algorithm with the minimum EER previously calculated is taken.

- The verification based on the biometric algorithm selected is performed and a match score is computed. The template related to the claimed identity is retrieved from the templates database and compared with the one extracted from the sample of the biometric trait just captured.
- The matching score is fused (as shown later) with others previously calculated (if any). If the fused score is greater than a threshold, then the process stops and the user is authenticated, otherwise the process continue with selecting the next best algorithm for the user, excluded those already used.
- If the last available algorithm is used, without having a final fused score greater than the threshold, then the user is rejected.

5.10.1. Fusion and normalization

As said above, the fusion at match-score level is the way we combine evidences coming from multiple matchers. This approach has the advantage of utilizing as much information as possible from each biometric modality, while at the same time enabling the integration of proprietary Commercial Off-the-Shelf (COTS) biometric systems (Neurotechnology in our case). Most vendors of biometric systems do not like to release the feature values computed by their systems. Moreover, among the categories available in such a level fusion, (i.e. classification-based, transformation-based and densities-based) we decided to focus on the transformation type.

When a match score is ready it is fused with others scores (if any) that have been calculated by other biometric algorithms, using the weighted sum rule. But before that, since multiple matchers could output score in different domains (different range of values or even complementary type of scores such as similarity/dissimilarity), a common domain and thus a normalization method is required. In literature a lot of normalization schemes have been proposed, among which we chose the Min-Max method. This is easy to implement but has shown good performance as well, together with the sum-rule fusion method. The Min-Max rule retains the original distribution of scores except for a scaling factor and transforms all the scores into the common range $[0, 1]$. Distance score can be transformed into similarity score by subtracting the min-max normalized score from 1. The normalized score is given by :

$$s' = \frac{s - min}{max - min}$$

Where $s^{'}$ is the normalized score, s is the score computed by the current biometric algorithm and max, min are the maximum and the minimum scores that current matcher can produce. If such values are not given by the producer, we can estimate them for a set of matching scores $\{s_k\}$[1]. The problem of this technique is that when the minimum and maximum are estimated, outliers will influence a lot the resulting performance. This method is in fact higly sensitive to infrequent high and low scores.

Having the scores from multiple matchers normalized in the same range, the fusion is made as follows :

$$S = \sum_{m=1}^{M} w_m \times s_m^{'}$$

Where M is the number of matchers, w_m is the weight associated to matcher m and $s_m^{'}$ is the normalized score coming from the matcher m. The weights w_m are calculated as follows :

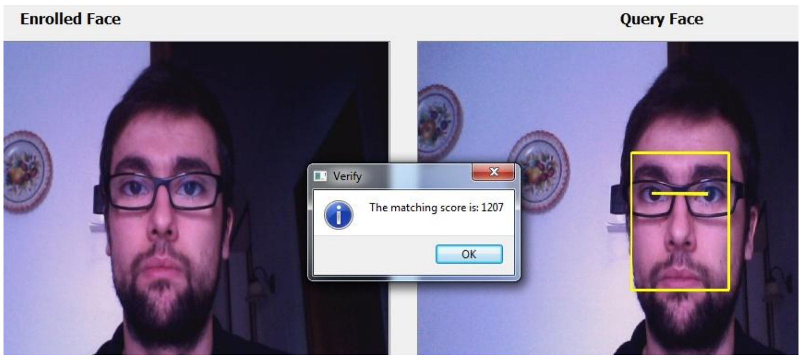
$$w_m = \frac{1/\sum_{m=1}^{M} \frac{1}{e_m}}{e_m}$$

Where e_m is the EER[2] of the matcher m, and $\sum_{m=1}^{M} w_m = 1$. Weights are inversely proportional to the corresponding errors; the weights for more accurate matchers are higher than those of less accurate matchers.

The fusion is executed just with the scores of algorithms that have already been used in the actual verification process, this because we do not always perform all the verification modalities, but the software decides at the end of each, if more information is needed (i.e. process another biometric trait) to take a final decision (e.g. when the first algorithm is used, then its normalized score $s_1^{'}$ is not fused with anyone else, when the second algorithm is used (if necessary) then its normalized score $s_2^{'}$ is fused with the $s_1^{'}$ using the sum rule, when the third algorithm is used (if necessary) then its normalized score $s_3^{'}$ is fused with $s_1^{'}$ and $s_2^{'}$, and so on). Below are some images of the authentication process and after the code through the verification process in the main program is implemented.

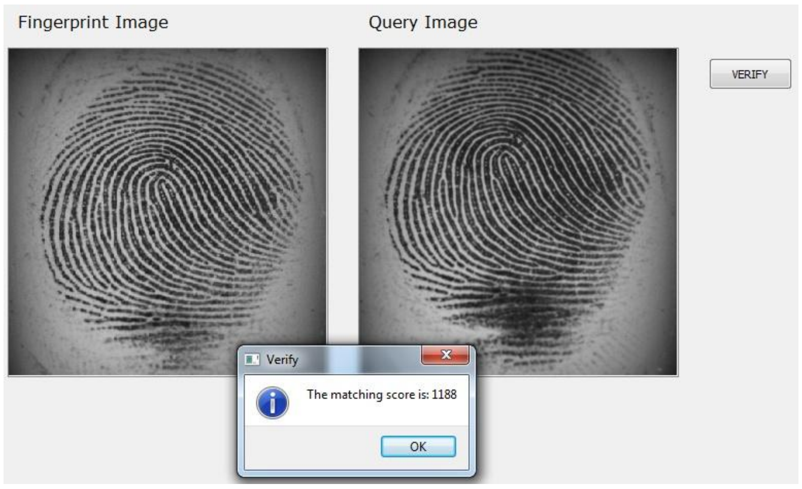
[1] In our case the minimum and maximum values are estimated using a database and performing all possible comparisons between images

[2] The EER of matcher m is calculated launching the function "test", available in the "InsertNewAlgorithm" window as explained later



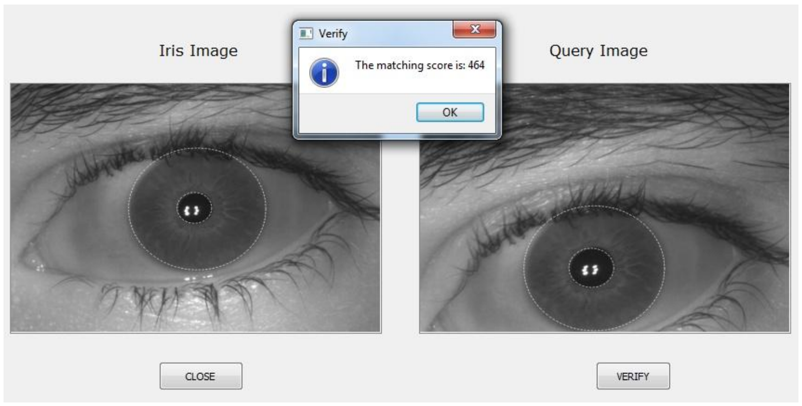
The matching score is: 1207

Figure 48 : Verification of the face



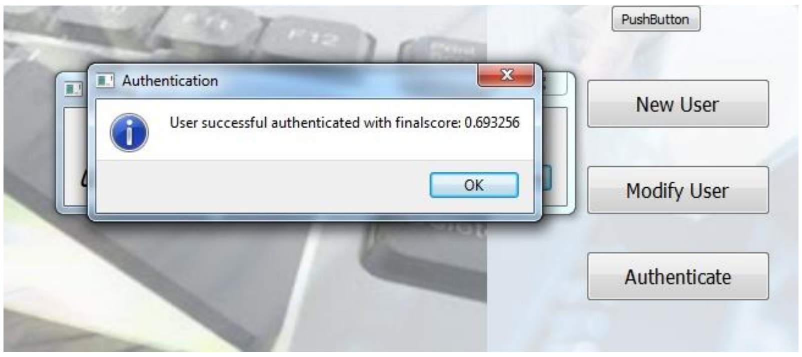
The matching score is: 1188

Figure 49 : Verification of the fingerprint



The matching score is: 464

Figure 50 : verification of the iris



User successful authenticated with finalscore: 0.693256

Figure 51 : final score given by the normalization and fusion of the three previous single scores

Code representing the call to the verify function of the algorithm, along with fusion and normalization of the scores

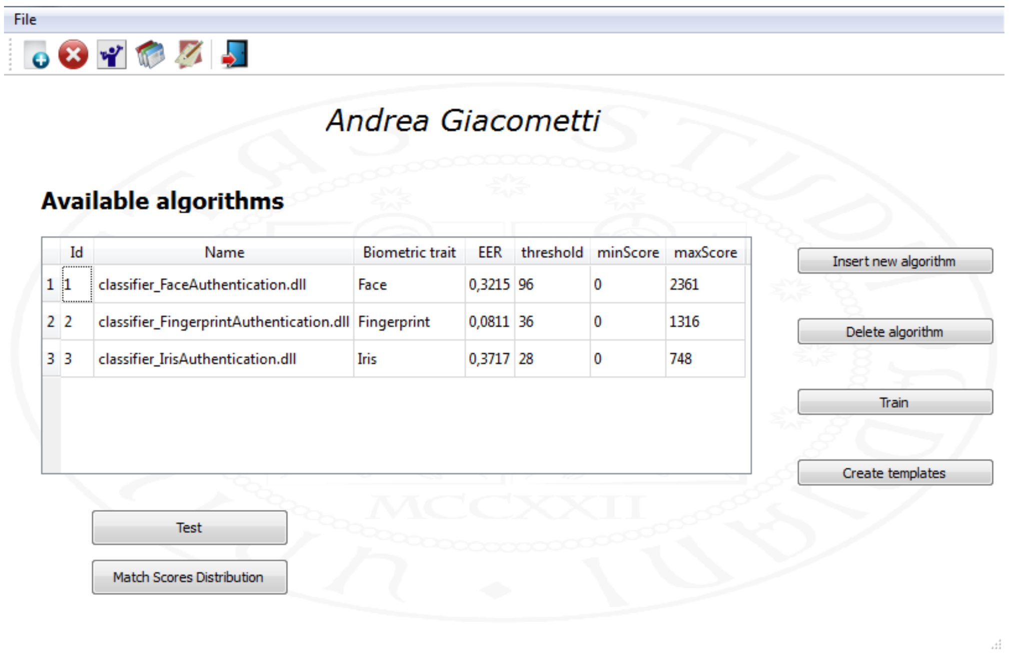
```
void AuthenticationDialog::verify(int userId)

{
    Result res;
    res = continueVerification(userId);

    while(continueWithOtherAlgorithm(res.decision, res.matchScore))
    {
        //get another algorithm and a new result
        res = continueVerification(userId);
    }

    if(finalResult.decision)
    {
        qDebug() << "Final result positive" << endl;
        QMessageBox::information(this, tr("Authentication"), QString("User
                                successful authenticated with final score:
                                %1").arg(finalResult.matchScore),
                                QMessageBox::Ok);
    }
    else
    {
        qDebug() << "Final result negative" << endl;
        QMessageBox::warning(this, tr("Authentication"), QString("The user has
                                not been authenticated since the final score is:
                                %1").arg(finalResult.matchScore),
                                QMessageBox::Ok);
    }
}



Result AuthenticationDialog::continueVerification(int userId)
{
    static bool firstVerification = true;
    qint64 timer = -1;
    Result res;
    res.decision = false;
    res.matchScore = 0;
    if(firstVerification)
    {
        getBestAlgorithm(userId);
        firstVerification = false;
    }
    QSqlQuery query(db);
    QString imagesPath, templatesPath;
    query.prepare("SELECT database_images, database_templates FROM traits WHERE
                    trait = :trait");
    query.bindValue(":trait", currentAlg->biometricTrait);
    query.exec();
    query.next();
    QString t = query.value(0).toString();
    QString p = query.value(1).toString();
    imagesPath = t.append(QString("/User_%1").arg(userId));
    templatesPath = p.append(QString("/User_%1/%2").arg(userId).arg(name));
    res = currentAlg->alg->verifyUser(imagesPath.toStdString(),
                                    static_cast<ostringstream*>( &(ostringstream()
                                    << userId) )->str(),
                                    templatesPath.toStdString());
    timer = currentAlg->alg->getVerifyTime();
    updateUserBestAlgorithm(userId);
    //min-max normalization

    double smin = currentAlg->sMin;
    double smax = currentAlg->sMax;
    if( smax != 0)
        res.matchScore =( res.matchScore - smin ) / smax;

    QString dateAndTime = QDateTime::currentDateTime().toString("yyyy/MM/dd
                                                                  hh:mm:ss");
    query.prepare("INSERT INTO statistics (user, algorithm, verify_time,
                  authentication_attempt_date, accepted)
                  "VALUES (:user, :algorithm, :verify_time,
                  :authentication_attempt_date, :accepted)");
    query.bindValue(":user", userId);
    query.bindValue(":algorithm", algorithmId);
    query.bindValue(":verify_time", timer);
    query.bindValue(":authentication_attempt_date", dateAndTime);
    query.bindValue(":accepted", (int)res.decision);
    bool r = query.exec();
    if(!r)
        QMessageBox::warning(this, tr("Database Error"),
                             query.lastError().text());

    usedAlgorithms.insert(currentAlg, res);
    return res;
}


void AuthenticationDialog::getBestAlgorithm(int userId)
{
    QListIterator<algorithm*> it(availableAlgorithms);
    algorithm* temp;
    QSqlQuery query(db);
    query.prepare("SELECT best_algorithm FROM users WHERE users.id = :id");
    query.bindValue(":id", userId);
    query.exec();
    query.next();
    //if the best-algorithm field was not initialized
    if(query.value(0).toInt() == 0)
    {
        double minEER = HUGE_VAL;
        while(it.hasNext())
        {
            temp = it.next();
            if(minEER > temp->EqualErrorRate)
            {
                minEER = temp->EqualErrorRate;
                currentAlg = temp;
                path = temp->path;
                name = temp->name;
                algorithmId = temp->id;
            }
        }
    }
    else
    {
        algorithmId = query.value(0).toInt();
        while(it.hasNext())
        {
            temp = it.next();
            if(temp->id == algorithmId)
            {
                currentAlg = temp;
                path = temp->path;
                name = temp->name;
                algorithmId = temp->id;
                break;
            }
        }
    }
    availableAlgorithms.removeOne(currentAlg);
}


bool AuthenticationDialog::continueWithOtherAlgorithm(bool previousDecision,
                                              double previousMatchingScore)
{
    double constant = 0;
    weight weights[usedAlgorithms.size()];
    algorithm* temp;
    QList<algorithm*> algorithms = usedAlgorithms.keys();
    QListIterator<algorithm*> it(algorithms);
    int i = 0;
    while(it.hasNext())
    {
        temp = it.next();
        constant = constant + 1/temp->EqualErrorRate;
        weights[i].e = temp->EqualErrorRate;
        weights[i].alg = temp;
        ++i;
    }
    constant = 1/constant;
    for(int i = 0; i < usedAlgorithms.size(); i++)
        weights[i].w = constant/weights[i].e;
    double totalScore = 0;
    for( int i = 0; i < usedAlgorithms.size(); i++)

        totalScore = totalScore + ( weights[i].w *
                                  usedAlgorithms.value(weights[i].alg).matchScore
);
    if(totalScore > Threshold)
    {
        finalResult.decision = true;
        finalResult.matchScore = totalScore;
        return false;
    }
    else //take another algorithm
    {
        finalResult.decision = false;
        finalResult.matchScore = totalScore;
        if(availableAlgorithms.size() != 0)
        {

            QListIterator<algorithm*> iterat(availableAlgorithms);
            double min = INFINITE;
            algorithm* temp;
            while(iterat.hasNext())
            {
                temp = iterat.next();
                if( min > temp->EqualErrorRate )
                {
                    min = temp->EqualErrorRate;
                    currentAlg = temp;
                }
            }
            algorithmId = currentAlg->id;
            name = currentAlg->name;
            path = currentAlg->path;
            availableAlgorithms.removeOne(currentAlg);
            return true;
        }
        else
            return false;
    }
}
```

5.11. Analyzing algorithms

When the software is started, a sequence of controls are made in order to check that all the database tables exist (if not they are created) required by modules that performs queries during the execution. Furthermore at the first execution it is asked to set :

- Which are the traits covered by algorithms that are available at that moment in the specific folder reserved for them (or by those algorithms that we are going to put in).
- For each biometric trait, specify the image and template database, where all the images and templates due to the enrollment of each user will be placed, following the scheme of section 5.9 (i.e. specify the location of the root showed in the scheme).

We can now start with inserting new algorithms in the database. Recall that they have to implement the “AlgorithmInterface”c lass, and has to be compiled as specified in section 5.7, in order to have DLLs ready to be dynamically loaded while the software runs.



Available algorithms

	Id	Name	Biometric trait	EER	threshold	minScore	maxScore
1	1	classifier_FaceAuthentication.dll	Face	0,3215	96	0	2361
2	2	classifier_FingerprintAuthentication.dll	Fingerprint	0,0811	36	0	1316
3	3	classifier_IrisAuthentication.dll	Iris	0,3717	28	0	748

Figure 52 : Screenshot showing the available algorithms and options to handle them

From the window shown in Figure 52, we can handle algorithms (delete, insert, train), and test them to find the minimum and maximum scores, FAR, FRR and EER that are used during verification process.

5.11.1. Test

The test function allows to define the selected algorithm's FAR and FRR (and thus the EER as well), to have an idea of the algorithm's accuracy. Selecting the algorithm in the available list, and pressing the Test button, a window is open in order to give the location of the database which will be used to test the algorithm (It is enough to select the root folder where the database is placed). The testing process works as follow :

- After giving the location of the relative root folder of the database, specify the number of characters that are useful to identify the subject.
  That is, since FAR and FRR are based on the results of genuine and impostor scores, the software needs to know when it is performing a genuine or an impostor comparison. This information is given by splitting the filename of the image in order to keep the part that identify the subject (that is the class it belongs to). Each image is then associated to that class and when compared with another one, the score obtained from the matcher is labeled genuine or impostor according to fact that the two images belong to the same class or not.
  As an example if in a fingerprint database the filenames have the xxxYYYzzz.jpg format, where xxx represents the person, YYY the finger and zzz the image number of the same finger, then the number of characters that identify a genuine comparison between an impostor one is 6.
  A recursive algorithm searches in all the subfolders, starting from the given one, looking for image files. All the image file paths are stored in a list along with the class they belong to, and then passed to the "test" method of the algorithm .
- The current Algorithm create a template for every image present in the list that received from the main program and save it in an array of templates. Then the matching threshold is set to 0 and each template is compared with all the others in the list. The output score is labeled as genuine if both templates belong to the same class (e.g. the same finger of the same person recalling the above example), as impostor otherwise. In the former case if the score is equal or lower than the threshold it is a False Reject case, in the latter case if the score is greater than the threshold it is a False Accept case.
  This step is repeated for each threshold value, creating an array of terns made by FAR, FRR and related threshold. This array are passed back to the main program.
- The main program receives the array containing all the FAR and FRR values corresponding to a certain threshold, and plot them making use of the Qwt libraries (see the code below

and Figure 53). It also updates the “farfrr” table with these values and calculate the EER, that is where FAR and FRR values has the minimum distance.

Code representing the plotting of the FAR-FRR curves given the vectors with FAR and FRR values for each threshold

```
void Graph::plotFARFRRcurve(QVector<double> &FalseAcceptanceRate,
                            QVector<double> &FalseRejectRate,
                            QVector<double> &Thresholds)
{
    // panning with the left mouse button
    (void) new QwtPlotPanner( canvas() );
    // zoom in/out with the wheel
    (void) new QwtPlotMagnifier( canvas() );
    setAutoFillBackground( true );
    setPalette( QPalette( QColor( 165, 193, 228 ) ) );
    updateGradient();
    //setTitle("A Simple QwtPlot Demonstration");
    insertLegend(new QwtLegend(), QwtPlot::RightLegend);
    // axes
    setAxisTitle(xBottom, "Threshold" );
    setAxisScale(xBottom, 0.0, 1.0);
    setAxisTitle(yLeft, "Percentage");
    setAxisScale(yLeft, 0.0, 1.0);
    // canvas
    canvas()->setLineWidth( 1 );
    canvas()->setFrameStyle( QFrame::Box | QFrame::Plain );
    canvas()->setBorderRadius( 15 );
    QPalette canvasPalette( Qt::white );
    canvasPalette.setColor( QPalette::Foreground, QColor( 133, 190, 232 ) );
    canvas()->setPalette( canvasPalette );
    // Insert new curves
    QwtPlotCurve *FAR = new QwtPlotCurve("FAR");
    FAR->setRenderHint(QwtPlotItem::RenderAntialiased);
    FAR->setLegendAttribute(QwtPlotCurve::LegendShowLine, true);
    FAR->setPen(QPen(Qt::red));
    FAR->setStyle(QwtPlotCurve::Lines);
    FAR->setCurveAttribute(QwtPlotCurve::Fitted, true);
    FAR->attach(this);
    QwtPlotCurve *FRR = new QwtPlotCurve("FRR");
    FRR->setRenderHint(QwtPlotItem::RenderAntialiased);
    FRR->setLegendAttribute(QwtPlotCurve::LegendShowLine, true);
    FRR->setPen(QPen(Qt::blue));
    FRR->setStyle(QwtPlotCurve::Lines);
    FRR->setCurveAttribute(QwtPlotCurve::Fitted, true);
    FRR->attach(this);
    QwtPointArrayData *serie1 = new QwtPointArrayData(Thresholds,
                                                      FalseAcceptanceRate);
    QwtPointArrayData *serie2 = new QwtPointArrayData(Thresholds,
                                                      FalseRejectRate);
    FAR->setData(serie1);
    FRR->setData(serie2);
    qSort(Thresholds.begin(), Thresholds.end());
    qSort(FalseAcceptanceRate.begin(), FalseAcceptanceRate.end());
    qSort(FalseRejectRate.begin(), FalseRejectRate.end());
    double xmin = Thresholds.first();
    double xmax = Thresholds.last();
    double ymax = FalseRejectRate.last() > FalseAcceptanceRate.last() ?
                  FalseRejectRate.last() : FalseAcceptanceRate.last();
    setAxisScale(yLeft, 0.0, ymax);
    setAxisScale(xBottom, xmin, xmax);
    QwtPlotMarker *mY = new QwtPlotMarker();
    mY->setLineStyle(QwtPlotMarker::HLine);
    mY->setYValue(0.0);
    mY->attach(this);
    QwtPlotMarker *cX = new QwtPlotMarker();
    cX->setLineStyle(QwtPlotMarker::VLine);
    cX->setXValue(0.0);
    cX->attach(this);
}
```

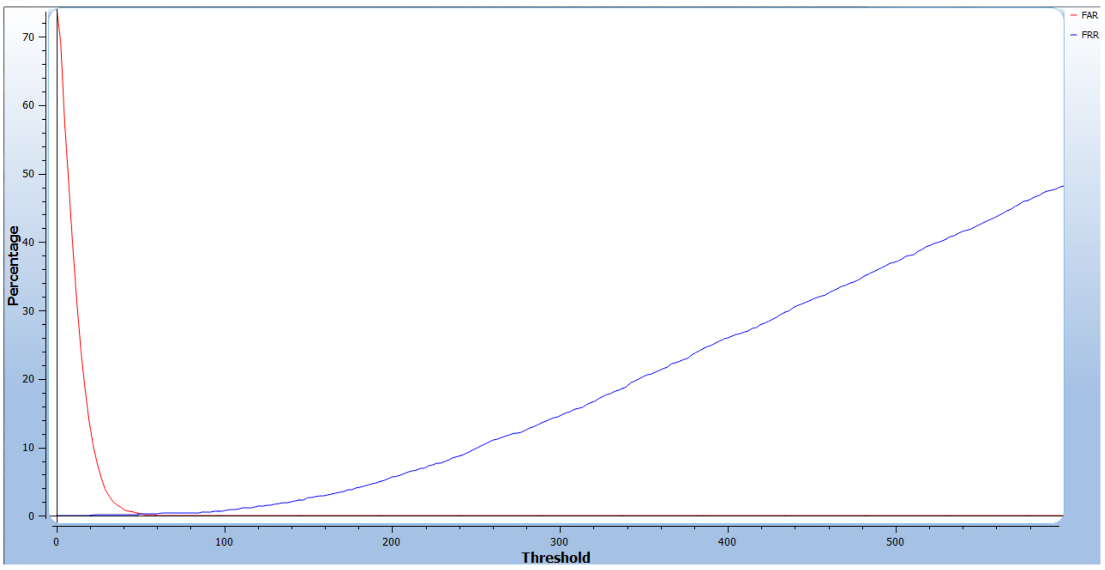


Figure 53 : FAR and FRR curves plotted against threshold values for the Face modality

5.11.2. Match score distributions

The “match score distributions” function is similar to the “test” function. Again for each database image a template is created and then compared with all the others. A vector of genuine scores is created and filled with values coming from genuine comparisons (two templates of the same class). The same for an impostor vector, filled with scores coming from impostor comparisons (two templates of different classes). Both vectors are then passed back to the main program and the same module used to plot the FAR and FRR curves is used to plot the matching score densities. But now some preliminary actions have to be done, such as find the minimum and maximum score values and update the “algorithms” table, calculate the relative frequencies after pointing out an appropriate bin width depending on the number of x points to plot the frequencies against and so on. An example of iris genuine and impostor plotting densities is shown in Figure 54.

It is important that the calculation of the individual algorithm's FAR, FRR, ERR, mimum and maximum values is done just after adding it to the available algorithms since they are required in verification phase to calculate the matching scores.

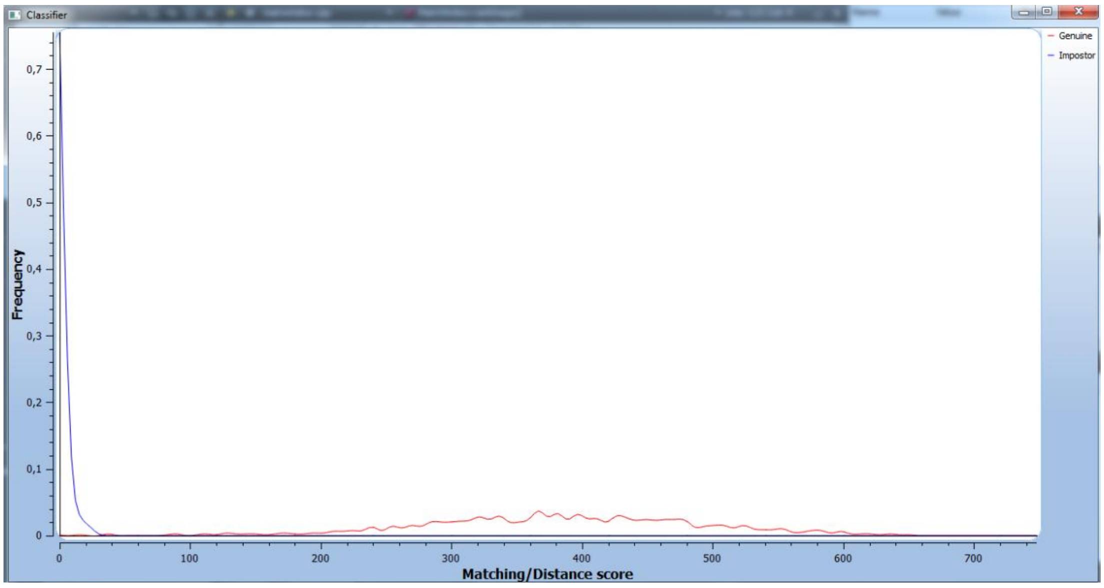


Figure 54 : Genuine and impostor matching score densities of the iris modality

6. Conclusions

A prototype software to integrate multi biometric evidences has been developed. The software does not use a fixed number of biometric algorithms to perform user authentication but changes dynamically. It also allows to add more algorithms in a later time, and even more biometric traits (ear recognition for example).

The software allows to enroll a new user creating and storing a template for each of user's biometric characteristic, and saving the image which has been extracted from. Then user data can be managed and changed.
A verification module permits to authenticate an individual, asking for his biometric traits one by one, that is deciding to get more biometric evidences (ask for another trait) if the amount of data gained so far is not enough to take a decision. The fusion is made thanks to previous score normalization and estimates of :

- Minimum and maximum score values of each matcher
- FAR, FRR and EER of each matcher

The software allows to calculate these parameters selecting the database we want to test the algorithms on, and results are plotted in a graph to get a visual feedback.

The fusion and normalization schemes are the sum-weighted and min-max normalization rules, respectively, since that in literature they continue to be often used in multibiometrics. In fact their performances confirm they still are among the most accurate techniques of fusion and normalization. Only in recent days combination of densities estimation techniques or SVM approaches are gaining attention due to high levels of accuracy.

The software automatically handles the underlying database, creating tables if they don't exist and synchronizing the MYSQL part with the information saved in the application folders.

As suggested from Figure 54, in the iris matcher case we do not have a significant overlap of the genuine and impostor regions (this is true for face and fingerprint matchers as well), meaning that it reaches an overall high accuracy (also suggested by a low value of EER). This was expected, given that COTS (Commercial Off-The-Shelf) systems leave less room for improvement.

About the face recognition module, we have seen that during tests, it was important that the environment illumination was quite homogeneous, otherwise the face template was not created and a "QualityCheckExposureFailed" (72) error was launched (for example when the light was coming mainly from one side).

At the end, no physiological characteristic is better than all the others in all the possible scenarios. The choice of which biometric traits to use in a biometric system is mainly dictated by the security requirements of the context it will work in, the available budget and the need or not of an identification module, since that it degrades the speed and the overall accuracy a lot. The increasing need of a simple and more secure way to get access in systems, such as ATMs, security areas, authentication for online payments and online services (mail accounts, health services), is giving more and more attention to the biometrics. But a potential very dangerous issue need to be taken into account, that is when the user's biometric data are stolen, then the thief would gain access to all the systems the user is registered to.

6.1. Future developments

Some future developments have been pointed out to enhance usability and the performance of this software :

- A multisamples approach could be integrated : during enrollment, the software could ask for more than one sample for each biometric characteristic and calculate an intra-class variation index to take into account when calculating, for example, the user-weights.
- Identification : an identification module could be added, recalling that the accuracy in this case is a significant issue, such as the speed performance.
- A normalization technique that reduces the outliers effect : if just one outlier value is found during test, can lead to make most of the data concentrate only in a small range during normalization. This is not unlikely to happen because some genuine users may get very high score when their present biometric data are compared to the data in the database. But most of users will not reach that high score for several reasons such as changes of the biometric trait, level of subject's cooperation, alteration of environment conditions and noise data. Removing outliers will increase the separation degree between genuine and impostor scores leading to a better performance.
- Testing the software : lack of time prevented us to exhaustively test the software. It will be necessary to do it in order to find hided weaknesses and system performance parameters.

Bibliography

1. [Online] http://www.neurotechnology.com/.

2. [Online] http://www.tweakandtrick.com/2012/06/most-common-authentication-methods-used.html.

3. **Lin, S.Y. Kung - M.W. Mak - S.H.** *Biometric Authentication - A Machine Learning Approach.* s.l. : Prentice Hall Professional Technical Reference, 2004.

4. Integrated Automated Fingerprint Identification System. *FBI - Federal Bureau of Investigation.* [Online] http://www.fbi.gov/about-us/cjis/fingerprints_biometrics/iafis/iafis.

5. Sydney Airport exapnds biometrics use. *COMPUTERWORLD the voice of IT management.* [Online] http://www.computerworld.com.au/article/372334/sydney_airport_expands_biometrics_use/.

6. Iris Scans at Amsterdam Airport Schiphol. *Schiphol Amsterdam Airport.* [Online] http://www.schiphol.nl/Travellers/AtSchiphol/Privium/Privium/IrisScans.htm.

7. **Zelazny, Frances.** The Evolution of India's UID Program . *Lessons Learned and Implications for Other Developing Countries.* s.l. : Center For Global Development, August 2012.

8. Privacy Impact Assessment for the Automated Biometric Identification System (IDENT). *Department of Homeland Security.* [Online] http://www.dhs.gov/xlibrary/assets/privacy/privacy_pia_usvisit_ident_final.pdf.

9. [Online] http://www.buzzle.com/articles/advantages-of-biometric-identification.html.

10. *Challenges and Constraints to the Diffusion of Biometrics in Information Systems.* **Akhilesh Chandra, Thomas Calderon.** 12, s.l. : Communications of the ACM, December 2005, Vol. 48.

11. **Nandakumar, Karthik.** A Dissertation. *Multibiometric Systems: Fusion Strategies and Tempate Security.* Michigan State University : s.n., 2008.

12. **Jain, Anil K.** Multimodal Biometric Systems. *Dept. of Computer Science and Engineering Michigan State University.*

13. Receiver operating characteristic. *Wikipedia, The Free Encyclopedia.* [Online] 03 25, 2013. [Cited: 03 25, 2013.] http://en.wikipedia.org/wiki/Receiver_operating_characteristic.

14. **KONUK, BARI.** Palmprint recognition based on 2-D GABOR filters. *A Thesis submitted to the Graduate School of Natural and apllied Sciences of Middle East Technical University.* January 2007.

15. **Methani, Chhaya.** Camera based palmprint recognition. *Master of Science (by Research) in Computer Science.* International Institute of Information Technology, Hyderabad, India : s.n., August 2010.

16. **Nicolò, Paganin.** Sistemi di Autenticazione Multibiometrica: Strumenti per il Deployment Automatico in Ambito Distribuito. *Master's Thesis, Padova University.* Padova : s.n., December, 2010.

17. **Nandakumar, Karthik.** Multibiometric Systems: Fusion Strategies and Template Security. *Submitted to Michigan State University.* 2008.

18. **Lin Hong, Anil Jain.** Integrating Faces and Fingerprints for Personal Identification. *Michigan State University.*

19. **Obied, Ahmed.** How to Attack Biometric Systems in Your Spare Time. *Department of Computer Science, University of Calgary, Alberta, Canada.* Calgary : s.n.

20. **Tsutomu Matsumoto, Hiroyuki Matsumoto, Koji Yamada, Satoshi Hoshino.** Impact of artificial "gummy" fingers on fingerprint systems. *Proc. SPIE 4677, Optical Security and Counterfeit Deterrence Techniques.* San Jose, CA : s.n., 2002. Vol. 4677.

21. **Xiaofu He, Yue Lu, Pengfei Shi.** A Fake Iris Detection Method Based on FFT and Quality Assessment. *CPR '08. Chinese Conference on Pattern Recognition.* 2008.

22. **Youngshin Kim, Jaekeun Na, Seongbeak Yoon, and Juneho Yi.** Masked fake face detection using radiance measurements. *J. Opt. Soc. Am.* 2009. Vol. 26.

23. **Arun Ross, Jidnya Shah, and Anil K. Jain.** From Template to Image: Reconstructing Fingerprints from Minutiae Points. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.* April 2007. Vol. 29, 4.

24. **Michelle Boatwright, Xin Luo.** What Do We Know About Biometrics Authentication. Virginia : s.n.

25. **Spadanuta, Karen Harmel and Laura.** Disney World scans fingerprint details of park visitors. *boston.com.* [Online] September 3, 2006. http://www.boston.com/news/nation/articles/2006/09/03/disney_world_scans_fingerprint_details_of_park_visitors/.

26. **Sergey Tulyakov, Venu Govindaraju.** Utilizing Independence of Multimodal Biometric Matchers. *Multimedia Content Representation, Classification and Security Lecture Notes in Computer Science.* 2006. Vol. 4105.

27. **Gupta, Badrinath G S and Phalguni.** An Efficient Multi-algorithmic Fusion System based on Palmprint fro Personnel Identification. *15th International Conference on Advanced Computing and Communications.*

28. **Jain, Lin Hong and Anil.** Integrating Faces and Fingerprints for Personal Identification. *Michigan State University, Department of Computer Science.*

29. **Yunhong Wang, Tieniu Tan, Anil K. Jain.** Combining Face and Iris Biometrics for Identity Verification.

30. **Nanni, Alessandra Lumini and Loris.** When Fingerprints Are Combined with Iris - A Case Study: FVC2004 and CASIA. *International Journal of Network Security PP.27-34.* 2007. Vol. 4, 1.

31. **Agrawal, Mohit.** Design Approaches for Multimodal Biometric System. *A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Technology.* August 2007.

32. **Ross, Arun.** An Introduction To Multibiometrics. *Proc. of the 15th European Signal Processing Conference (EUSIPCO), (Poland).* Poznan : s.n., September 2007.

33. **Paliwal, C. Sanderson and K. K.** Information Fusion and Person Verification Using Speech and Face Information. *Research Paper IDIAP-RR 02-33, IDIAP.* September 2002.

34. **B. Ulery, A. R. Hicklin, C. Watson, W. Fellner, and P. Hallinan.** Studies of Biometric Fusion. *Technical Report IR 7346, NIST.* September 2006.

35. **Mingxing He, Shi-Jinn Horng, Pingzhi Fan, Ray-Shine Run, Rong-Jian Chen, Jui-Lin Lai, Muhammad Khurram Khan, Kevin Octavius Sentosa.** Performance evaluation of score level fusion in multimodal biometric systems. *Article in Press, Pattern Recognition.* November 2009.

36. **Arun Ross, Anil Jain.** Information fusion in biometrics. *Pattern Recognition Letters 24.* 2003.

37. **J. Bigun, J Fierrez-Aguilar, J. Ortega-Garcia, J. Gonzalez-Rodriguez.** Multimodal Biometrich Authentication using Quality Signals in Mobile COMMUNICATIONS. *in Proc-ICIAP 2003.*

38. **C. Andreson, K.K. Paliwal.** Information Fusion and Person Verification Using Speech & Face Information". *IDIAP Research Report 02-33.* September 2002.

39. **Sonia Garcia-Salicetti, Mohamed Anouar Mellakh, Lorène Allano, Bernadette Dorizzi.** Multimodal Biometric Score Fusion: the Mean Rule VS. Support Vector Classifiers. *Department Electronique et Physique Institut National des Telecommunications.*

40. **Yu.A. Zuev, S.K. Ivanov.** The voting as a way to increase the decision reliability. *Journal of the Franklin Institute 336.* 1999.

41. **Kuncheva, L. I.** Combining Pattern Classifiers - Methods and Algorithms. s.l. : Wiley, 2004.

42. **Lei Xu, Adam Krzyzak and Ching Y. Suen.** Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS.* June 1992. Vol. 22, 30.

43. **Anil Jain, Karthik Nandakumar, Arun Ross.** Score normalization in multimodal biometric systems. *Pattern Recognition 38.* 2005.

44. **Robert Snelick, Umut Uludag, Alan Mink, Michael Indovina, Anil Jain.** Large Scale Evaluation of Multimodal Biometric Authentication using State-of-the-Art Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* Mar 2005. Vol. 27, 3.

45. **Zahid Akhtar, Giorgio Fumera, Gian Luca Marcialis, Fabio Roli.** Evaluation of Multimodal Biometric Score Fusion Rules under Spoof Attacks. *Department of Electrical and Electronic Engineering - University of Cagliari.* Cagliari : s.n.

46. **Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas.** On Combining Classifiers. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.* March 1998. Vol. 20, 3.

47. **R. Snelick, M. Indovina, J. Yen, A. Mink.** Multimodal biometrics: issues in design and testing. *Proceedings on Fifth International Conference on Multimodal Interfaces.* Vancouver, Canada : s.n., 2003.

48. **Karthik Nandakumar, Yi Chen, Sarat C. Dass and Anil K. Jain.** Likelihood Ratio-Based Biometric Score Fusion. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.* February 2008. Vol. 30, 2.

49. **Raghavendra R, Rao A, Hemantha Kumar G.** Multimodal biometric score fusion using Gaussian mixture model and Monte Carlo Method. *JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY.* July 2010. Vol. 25.

50. **Sarat C. Dass, Karthik Nandakumar and Anil K. Jain.** A principled Approach to Score Level Fusion in Multimodal Biometric Systems. *To appear in Proceedings of AVBPA.* 2005.

51. **Fawaz Alsaade, Aladdin Ariyaeeinia, Amit Malegaonkar, Surosh Pillay.** Qualitative fusion of normalised scores in multimodal biometrics. *Pattern Recognition Letters.* 2009. Vol. 30.

52. **Gaurav Aggarwal, Nalini K. Ratha, Ruud M. Bolle and Rama Chellappa.** MULTI-BIOMETRIC COHORT ANALYSIS FOR BIOMETRIC FUSION. 2008.

53. **K.Sasidhar, Vijaya L Kakulapati, Kolikipogu Ramakrishna and K.KailasaRao.** MULTIMODAL BIOMETRIC SYSTEMS – STUDY TO IMPROVE ACCURACY AND PERFORMANCE. *International Journal of Computer Science & Engineering Survey.* 2010. Vol. 1, 2.

54. **Mehdi Parviz, M. Shahram Moin.** Boosting Approach for Score Level Fusion in Multimodal Biometrics Based on AUC Maximization. *Journal of Information Hiding and Multimedia Signal Processing.* January 2011. Vol. 2, 1.

55. Face_detection. *Wikipedia.* [Online] [Cited: 04 6, 2013.] http://en.wikipedia.org/wiki/Face_detection.

56. **Paul Viola, Michael Jones.** Rapid Object Detection using a Boosted Cascade of Simple Features. *ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION.* 2001.

57. How Face Detection Works. *Cognotics - Resources for Cognitive Robotics.* [Online] http://www.cognotics.com/opencv/servo_2007_series/part_2/sidebar.html.

58. **Sanjay Kr. Singh, D. S. Chauhan, Mayank Vatsa, Richa Singh.** A Robust Skin Color Based Face Detection Algorithm. *Tamkang Journal of Science and Engineering.* 2003. Vol. 6, 4.

59. **Turk, Sheng Zhang and Matthew.** Eigenfaces. *Scholarpedia.* [Online] [Cited: 01 08, 2013.] http://www.scholarpedia.org/article/Eigenfaces.

60. **Pentland, Matthew Turk and Alex.** Eigenfaces for recognition. *Journal of Cognitive Neuroscience.* 1991. Vol. 3, 1.

61. **Rao, A.R.** A Taxonomy for texture description and Identification. New York : Springer-Verlack, 1990.

62. **Anil Jain, Lin Hong and Ruud Bolle.** On-Line Fingerprint Verification. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE.* April 1997. Vol. 19, 4.

63. **Michael Boyd, Dragos Carmaciu, Francis Giannaros, Tom Payne and William Snell.** Iris Recognition Application. *Project Iris.* [Online] http://projectiris.co.uk/final.pdf.

64. **Daugman, John.** How Iris Recognition Works. *IEEE Transactions on circuits and systems fro video technology.* 2004. Vol. 14, 1.

65. VistaFA2 Single Iris & Face Camera. *vistaimaging.com.* [Online] http://www.vistaimaging.com/FA2_product.html.

66. Fx3100 - Fingerprint scanner Match on Board. *biometrika.it.* [Online] http://www.biometrika.it/eng/fx3100.html.

67. **Locascio, Francesco.** Sistemi di Autenticazione Multibiometrica: strumenti a supporto della progettazione. *Matser's thesis, Padova University.* Padova : s.n., December 2010.

68. pam-face-authentication. *code.google.com.* [Online] https://code.google.com/p/pam-face-authentication/.

69. *neurotechnology.com.* [Online] http://www.neurotechnology.com/.

70. **Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides.** *Design Patterns - Elements of Reusable Object-Oriented Software.* s.l. : Addison-Wesley Professional, 1994.

71. **Lorene Allano, Sonia Garcia-Salicetti and Bernadette Dorizzi.** An Adaptive Multi-biometric Incremental Fusion Strategy in the Context of BMCE 2007. 2007.

72. **Neurotechnology.** *MegaMatcher 4.4, VeriFinger 6.6, VeriLook 5.3, VeriEye 2.6 and VeriSpeak 1.2 SDK - Developer's Guide.*

73. Qt SDK. *qt.digia.com.* [Online] http://qt.digia.com/Product/Qt-SDK/.

74. Download Qt, the cross-platform application framework. *qt-project.org.* [Online] http://qt-project.org/downloads.