Contract Number:	IST-2002-507634
Project Acronym:	BioSecure
Project Title:	Biometrics for Secure Authentication
Instrument:	Network of Excellence
Start Date of Project:	01 June, 2004
Duration:	36 months

Deliverable Number:	D2.2.2
Title of Deliverable:	Description and documentation of the BioSecure software library
Contractual Due Date:	31 December, 2006
Actual Date of Completion:	23 April, 2007
Workpackage contributing to the Deliverable:	WP-A2.2 – Development and test of open source reference systems
Nature of the Deliverable: (R/P/D/O)*	P
Lead Contractor for this Deliverable :	GET (partner 2)
Author{s}:	Dijana Petrovska and Aurélien Mayoue

*Nature: R-report, P-prototype, D-demonstrator, O-other

Abstract: This document introduces the BioSecure benchmarking reference systems and their associated documentation.

The document serves as supporting documentation and information reference for the “prototype deliverable”.

Keyword List: Biometrics: 2D and 3D face, speech, talking face, dynamic signature, iris, fingerprint, hand modality; multimodality; performance evaluation of biometric systems; reference biometric systems

Project co-funded by the European Commission within the sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

DOCUMENT EVOLUTION

Version	Date	Status	Notes
a0	13/12/06	QA specification	Content defined
a1	25/01/07	1st draft	Drafted by D. Petrovska
a2	19/03/07	2nd draft	Contribution from A. Mayoue included. Submitted to Activity A2 for comments.
b3	11/04/07	revised version	Approved by Activity A2. Version submitted to the Steering Board.
b4	23/04/07	final version	Approved version to provided to the Commission

EXECUTIVE SUMMARY

This deliverable describes the benchmarking reference systems and the associated documentation which constitutes the BioSecure Reference and Evaluation Framework. It should be noted that this framework consists of state-of-the-art or baseline open-source "reference" software, each program being accompanied with at least one publicly available database and well defined evaluation protocols. The benchmarking reference systems are either pre-existing ones, or those which have been developed by BioSecure partners during this project.

This benchmarking framework is mainly a result of the common work performed within activities A1, A2, A7 and A8. The very first version of the BioSecure framework was successfully tested during the 1st BioSecure Residential Workshop, held in Paris during Summer 2005.

In this deliverable, the current status of the BioSecure Reference and Evaluation Framework is provided. In the introduction, the motivation behind the need for such a framework is discussed. Then, the main components of the framework are summarized. The address of the server, on which these components are currently available, is given. Finally, some typical examples of documents included in the framework and some screenshots of what can be found on the server, are presented.

CONTENTS

1 INTRODUCTION

When addressing the problems of identity verification, it is often noticed how it is very difficult to compare two different methods from two different papers in the literature, even though they deal with the very same task. This raises a real problem when one wants to know, for example, if a new original approach performs better than the current state of the art. This can be explained by the fact that many researchers have recorded their own test database and are the only ones performing experiments on it, which are subsequently impossible to reproduce.

The benchmarking reference systems bring an easy yet efficient answer to this problem. Since such systems are open-source and freely available for everybody, when publishing results on a specific database according to a specific protocol, experiments using a reference system can be considered as a way of calibrating the difficulty of a particular task.

Furthermore, the benchmarking reference systems are made of replaceable modules, which is also of great interest. Indeed, researchers often work on a specific part of a system and do not have necessarily time nor interest in building a complete system. In this way, for example, a researcher could demonstrate the improvement of his new features extraction algorithm simply by replacing the corresponding module in the reference system, without having to bother about the pattern recognition algorithm.

In the BioSecure Reference and Evaluation Framework, each reference system is accompanied with a documentation which explains how to install and use this system. The documentation describes also a publicly available database and defines an evaluation protocol (that are defined respectively as the reference database and the reference protocol). Finally, the documentation provides the results obtained on the reference database according to the reference protocol. In this way, the documentation should allow full reproducibility of the benchmarking framework results.

To sum up, the main properties of a benchmarking reference system are:

- absolute reproducibility: a reference system is accompanied with one publicly available database and evaluation protocols;
- development as open-source software;
- easy implementation: in general, a reference system is composed of four modules (pre-processing, feature extraction, model building and matching).

2 THE BIOSECURE REFERENCE AND EVALUATION FRAMEWORK

The BioSecure Reference and Evaluation Framework consists of twelve benchmarking reference systems (open-source software libraries) whose main characteristics are reminded below:

- 2D face reference system: baseline system developed by Bogazici University (BU) and based on the standard eigenface approach (PCA).
- 3D face reference system: comparison system developed by Bogazici University (BU). It is based on Iterative Closest Point (ICP) and Thin Plate Spline (TPS) warping algorithms.
- Fingerprint reference system: link to NIST software which is based on a standard minutiae approach.
- Hand reference systems: two comparison systems. The system developed by GET is based on the geometry of the fingers whereas the system developed by BU is based on the form and the texture of the hand.
- Iris reference system: baseline system developed by GET and inspired by Daugman works.
- Signature reference systems: three comparison systems. The system developed by GET is based on HMM whereas the systems developed by Magdeburg University and UniKent are based on distance measures.
- Speech reference systems: links to state-of-the-art ALIZE and BECARS software which are based on GMM.
- Talking-face: comparison system developed by GET and based on the fusion of face and speech modality.

The BioSecure reference systems are currently available on a server (hosted by GET-INT) at the following address:

http://share.int-evry.fr/svnview-eph/

It is noticed that this address is not to be widely used until the final implementation of an open source license approach (LGPL).

For the moment, there is only one reference system per modality (2D-face, fingerprint, hand, iris, speech, signature and talking-face). These reference systems have been tested. In order to allow reproducibility of the benchmarking results for each system, there is a documentation ("how-to") which explains how to install and run the system with the reference database and protocol that have been defined.

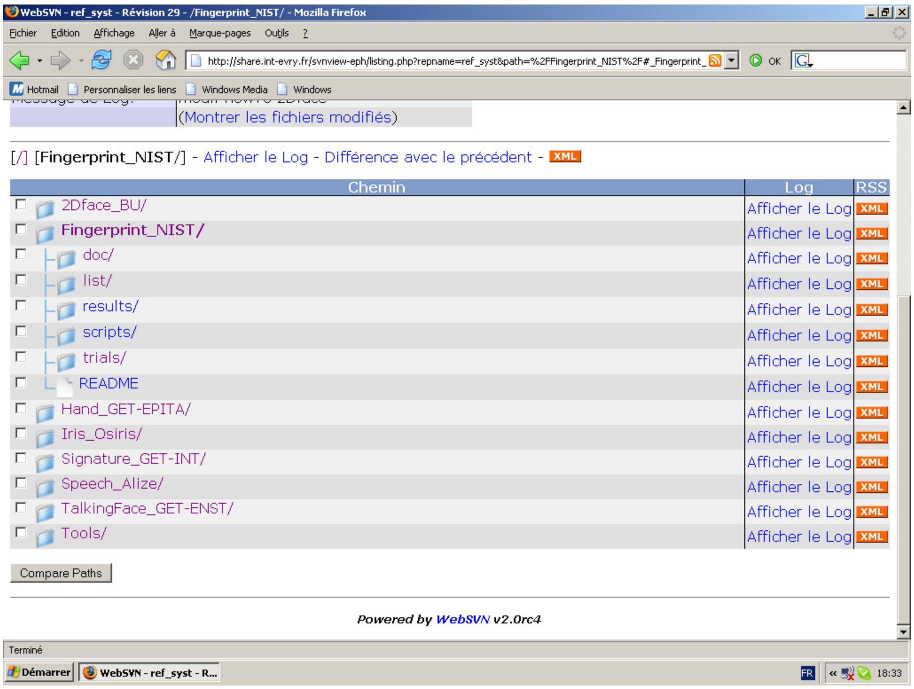
When a reference system is put on the server, each time the same sub-folders are created in order to have a common structure. The different parts of this structure are described hereafter:

- src: source code of the reference system / software library;
- doc: documentation which explains how to install and run the reference system on the reference database according to the reference protocol;
- results / scores: intermediate results and final scores (obtained on the reference database in accordance with the reference protocol);
- scripts: scripts required to run a full experiment;
- trials: list of (genuine and impostor) trials (according to the reference protocol);
- tar_files: required libraries;
- download: reference system / software library in the form of compressed file (tar.gz).

3 EXAMPLES

The material available (on the server) for each modality is not the same according to whether the reference system has been developed by the BioSecure members or not.

When open-source systems were already developed (such as the NIST programs for fingerprints or the ALIZE open-source toolkit for speaker recognition), BioSecure added publicly available databases and reference protocols to the open-source programs. This should allow the reproducibility of the benchmarking framework results. In this case, it shall be noticed that the source code of the reference system is not available on the server but the documentation "how-to" (included in the sub-folder *doc*) gives a link to the system. This explains why in such a case sub-folders *src* and *download* are not available on the server (for example, see figure 1 which shows what can be found on the server for the fingerprint modality).

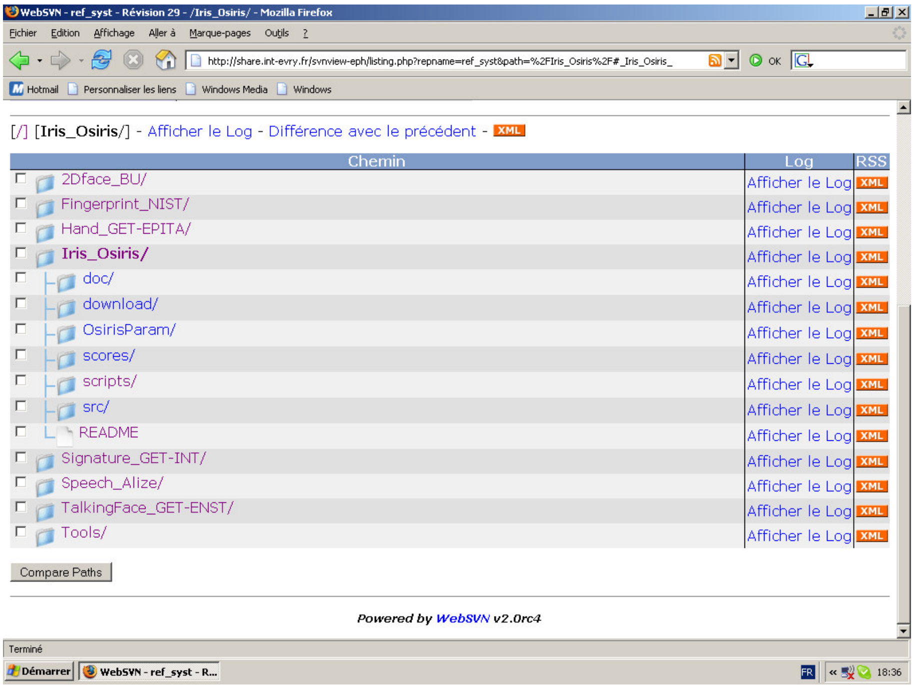


[/] [Fingerprint_NIST/] - Afficher le Log - Différence avec le précédent - XML

Chemin	Log	RSS
☐ 2Dface_BU/	Afficher le Log	XML
☐ Fingerprint_NIST/	Afficher le Log	XML
☐ doc/	Afficher le Log	XML
☐ list/	Afficher le Log	XML
☐ results/	Afficher le Log	XML
☐ scripts/	Afficher le Log	XML
☐ trials/	Afficher le Log	XML
☐ README	Afficher le Log	XML
☐ Hand_GET-EPITA/	Afficher le Log	XML
☐ Iris_Osiris/	Afficher le Log	XML
☐ Signature_GET-INT/	Afficher le Log	XML
☐ Speech_Alize/	Afficher le Log	XML
☐ TalkingFace_GET-ENST/	Afficher le Log	XML
☐ Tools/	Afficher le Log	XML

Figure.1 – Screenshot of what can be found on the server for the fingerprint modality

For a number of modalities, when no existing open-source software was available, the programs were developed within BioSecure. In that case, the source code of the programs is made available on the server (for example, see figure 2 which shows what can be found on the server for the iris modality). In that case, a publicly available database and reference protocols are equally defined to allow the complete reproducibility of the benchmarking framework results.



[/] [Iris_Osiris/] - Afficher le Log - Différence avec le précédent - XML

Chemin	Log	RSS
☐ 2Dface_BU/	Afficher le Log	XML
☐ Fingerprint_NIST/	Afficher le Log	XML
☐ Hand_GET-EPITA/	Afficher le Log	XML
☐ **Iris_Osiris/**	Afficher le Log	XML
☐ ├ doc/	Afficher le Log	XML
☐ ├ download/	Afficher le Log	XML
☐ ├ OsirisParam/	Afficher le Log	XML
☐ ├ scores/	Afficher le Log	XML
☐ ├ scripts/	Afficher le Log	XML
☐ ├ src/	Afficher le Log	XML
☐ └ README	Afficher le Log	XML
☐ Signature_GET-INT/	Afficher le Log	XML
☐ Speech_Alize/	Afficher le Log	XML
☐ TalkingFace_GET-ENST/	Afficher le Log	XML
☐ Tools/	Afficher le Log	XML

Compare Paths

Powered by WebSVN v2.0rc4

Figure 2 – Screenshot of what can be found on the server for the iris modality

In both cases, a detailed description of how to run each reference system is given in the documents present on the server. For example, the document “iris-osiris-howto” (respectively “fingerprint-nist-howto”) gives detailed guidelines of the steps that have to be done in order to run a full benchmarking experiment for iris (respectively fingerprint) with the OSIRIS (respectively NIST) benchmarking framework. These two documents are successively reproduced hereafter as annexes A and B.

ANNEX A

GUIDELINES TO RUN A BENCHMARKING EXPERIMENT FOR IRIS

A biometric reference system for iris

OSIRIS version 1.0

`howto last updated - 21/03/2007`

Foreword. *The reference system was developed by GET-INT (Emine Krichen, Anouar Mellakh, Sonia Salicetti and Bernadette Dorizzi). The following 'how to' was written by Aurélien Mayoue (GET-INT).*

Abstract. *The OSIRIS (Open Source for IRIS) reference system is an open source iris recognition system developed in the framework of the BioSecure project. The system is inspired by Daugman works [1]. It is composed of a segmentation and classification steps. The segmentation part uses the canny edge detector and the cicular Hough transform to detect both iris and pupil. The classification part is based on Gabor phase demodulation and Hamming distance classification.*

Introduction

This document describes the OSIRIS reference system and explains how to use it in practical to make an automatic verification experiment based on iris modality. This document is divided into four parts. In the first part, we present how to install the system. Then, in the second part, the database used and the protocol which have been defined are described. In the third part, we move on to the explanation of each module and finally, the experimental results obtained in authentication are shown in the last part.

I.Installation

The full system has been tested under Linux (Red Hat 3.2.3-42). For the compilation step, we have used g++ 3.2.3.

1.Downloading

The source files are currently available on the INT subversion server at the following address :

https://share.int-evry.fr/svn/eph/ref_syst/Iris_Osiris/

To download the source files in your home directory */home*, you have to use the 'svn' command as follows :

```
>svn checkout https://share.int-evry.fr/svn/eph/ref_syst/Iris_Osiris/
```

A succesful download creates the folder *Iris_Osiris* in your home directory.

2.Compilation

To compile the source code, you have to follow the next steps :

- enter in the *src* directory :
```
>cd Iris_Osiris/src
```
- compile the source code :
```
>make all
```

This compilation step creates the executable `osiris` in your directory */home/Iris_Osiris/src.*

II.Reference database and protocol

1.Reference database : Casia-BioSecure

The Casia-BioSecure (CBS) iris image database is a combination between two databases : Casia v2 and BioSecure v1 which were acquired following the same acquisition protocol.
Two kinds of cameras were used to construct the iris database : the OKI device and the PATTEK device. The OKI device is a hand-held camera produced by OKI, whose size is small and is suitable for small-scale personal verification, such as PC login, e-commerce, information security. The distance between the camera and the eye is about 4cm. The PATTEK device was developed by CASIA for physical access control in large buildings or for border control. The distance between the camera and the eye is about 10cm.
For each camera and for each person, twenty good quality iris images are selected automatically from a 35 second video. For the subjects wearing eyeglasses, both the iris images with/without eyeglasses will be captured (in this case, the 10^{th} first images will be acquired without eyeglasses whereas the 10^{th} remaining images will be acquired with eyeglasses). Temporal variability is not present in the reference database.
Finally, for each device, the CBS database includes 2400 iris images from 120 eyes with some intra class variability including illumination, glasses, eyelids/eyelashes occlusion, blurred and noisy images (see table (1)). Furthermore, the CBS database is the first iris database with comparable number of Asian and European persons.

databases	**devices**	**number of subjects**	**number of eyes**	**number of iris images**
Casia v2	OKI	30	60 $(2 * 30)$	1200 $(20 * 60)$
	PATTEK	30	60 $(2 * 30)$	1200 $(20 * 60)$
BioSecure v1	OKI	30	60 $(2 * 30)$	1200 $(20 * 60)$
	PATTEK	30	60 $(2 * 30)$	1200 $(20 * 60)$

Tab. 1 – description of the reference databases

2.Reference protocol

The data acquired with the two different devices (OKI and PATTEK) will not be mixed. So, two experiments will be carried out :

- an experiment with data acquired using the OKI device.
- an experiment with data acquired using the PATTEK device.

For each experiment, data consists of 2400 iris images from 120 eyes. The protocol defined for both experiments is the same. We divided datasets into two different sets :

- a training dataset composed by the 10^{th} first images of each person.
- a test dataset composed by the 10^{th} remaining images.

This protocol leads to test images acquired under different illuminations and compared images acquired with and without wearing eyeglasses effects.
For intra class comparisons, the 10 images of the training dataset are compared to the 10 images of the test dataset.
For inter class comparisons, the 10 images of the training dataset are compared to 10 images which are randomly selected to be considered as impostors.
Therefore, the total number of genuine and impostor trials is the same and is equal to $10 \times 10 \times 120 = 12000$, for each device.

III.Presentation of the software

The reference system has several sub-modules, which are described in this part. The execution of each sub-module is based on the use of an executable (`osiris`) and/or a script which is made of 7 fields. In the second case, the user will have to fill in these 7 fields which contain the same pieces of information for each sub-module :

1. Options : options required by the executable `osiris`.
2. Filter and Points : path to the text filter file and the text points file. These two files are needed for the features extraction (see part III.3) and are respectively located at */home/Iris_Osiris/OsirisParam/GaborFilterBank.txt* and */home/Iris_Osiris/OsirisParam/Point.txt*.
3. Path_source : path to the directory where the input files are located.
4. Path_results : path to the directory where the output file will be stored.
5. Result_file : name of the text results file. This file is only produced at the matching step (see part III.3).
6. Match : field containing the names of input files which are processed by the sub-module.

N.B : even if all these pieces of information are not needed by the sub-module, all fields have to be filled in for the good work of the script.

1.Segmentation

a.Description

The sub-module *segmentation* aims at isolating the iris from the eye image. To do that, it localizes both inner and upper boundaries of the iris rim by the Hough transform (these boundaries are assumed to be circular). This module was optimized for the Casia v1 database images and works only with iris images in BMP format (if you use images in other format, you will have to convert them before using this sub-module).

b.Software presentation

Use : `./osiris -d -i -l -R path/image.bmp`

Input : `path/image.bmp` is the path to the original eye image (*.bmp*). This path is specified with the standard option `-R`. Two original eye images from CBS database are shown in figure 1.

[a] [b]

Fig. 1 – Example of original eye images from Casia v2 [a] and BioSecure v1 [b] (these images were aquired using the OKI device)

Options :

1. `-d` : only segmentation is performed on the original eye image `path/image.bmp`.
2. `-i` : the output segmented image is saved as `path/image_Seg.bmp`.
3. `-l` : the parameters of the segmentation are saved in a file called `path/image.Param`.

Output : The sub-module creates two output files which are stored in the same directory as the one of the original eye image :

1. the parameters file (`path/image.Param`) which contains the following information (in order) :
   - information about the inner boundary (circle) of the iris rim : x and y coordinates of the center and radius of the circle.
   - information about the upper boundary (circle) of the iris rim : x and y coordinates of the center and radius of the circle.

An example of a such file is given in figure 2.

```
XPUP : 316
YPUP : 255
RPUP : 49

XIRIS : 321
YIRIS : 260
RIRIS : 120
```

Fig. 2 – An example of a parameters file produced by the module *segmentation*

2. the segmented image (`path/image_Seg.bmp`) which illustrates what the sub-module has done. It corresponds to the original eye image on which both inner and upper boundaries of the iris rim have been drawn (using information from the parameters file). An example of a such image is given in figure 3.

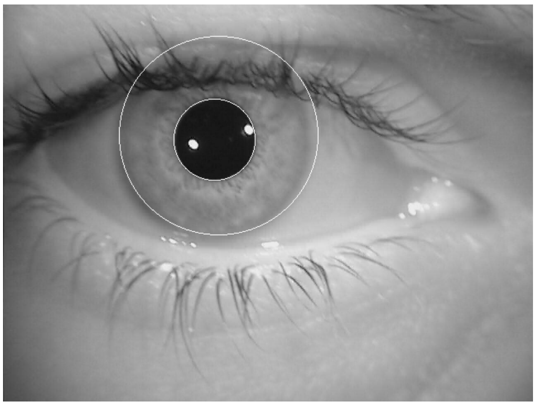


Fig. 3 – An example of a segmented image produced by the module *segmentation*

2.Normalisation

a.Description

The sub-module *normalisation* aims at normalizing the iris area in terms of size and illumination.

b.Software presentation

Use : `./osiris -s script`

The execution of the sub-module is based on the use of a script (see the introduction of the part III for a global description of a such script) whose the path (`script`) is specified with the standard option `-s`. The fields of this script used by the sub-module are described below :

Path_source : path to the directory which contains the input files, i.e. the original eye images (*.bmp*) and their parameters files (these latest files are produced by the previous sub-module).

Match : on each line of this field, there are two image filenames followed by their corresponding parameters filenames (each filename is separated by the tabulation character). An illustration of this field is shown in figure 4.

```
Match :0000_000.bmp    0000_001.bmp   0000_000.Param   0000_001.Param
Match :0000_002.bmp    0000_003.bmp   0000_002.Param   0000_003.Param
Match :0000_004.bmp    0000_005.bmp   0000_004.Param   0000_005.Param
...
```

Fig. 4 – Illustration of the field `Match` required by the module *normalisation*

Options :

1. `-k` : only normalisation is performed.
2. `-p` : the parameters files are taken into account for the normalisation step.
3. `-w` : from an input image called `image.bmp`, the normalized image will be saved as `image_Norm.bmp`.

Path_results : path to the directory which contains the output files, i.e. the normalized images of iris (*.bmp*). An example of a such image is given in figure 5.

Fig. 5 – An example of a normalized image (512 × 64 pixels) produced by the module *normalisation*

3.Feature extraction and Matching

a.Description

The sub-module *feature extraction* performs a convolution operation between the normalized iris image and a set of Gabor filters at pre-fixed points. At the end of this operation, we have at disposal a set of coefficients representing the iris. Each coefficient is then coded depending on its sign, resulting in a binary code of fixed length (denoted as iris code).

The sub-module *matching* provides a score. A comparison between two iris codes using the Hamming distance is performed. The score is included between 0 and 1, and it equals 0 when the two iris codes are strictly the same.

b.Software presentation

Use : `./osiris -s script`

The execution of the two sub-modules is based on the use of an only script (see the introduction of the part III to have a global description of a such script) whose the path (`script`) is specified with the standard option `-s`. The fields of this script used by the sub-modules are described below :

Path_source : path to the directory which contains the input files, i.e. the normalized images of iris (*.bmp*) which are provided by the previous sub-module (*normalisation*).

Filter : path to the filter file (*.txt*) which contains a set of Gabor filters.

Points : path to the points file (*.txt*) containing the coordinates of points at which the convolution between the normalized image and the set of filters is performed.

Match : on each line of this field, there are the filenames of the two normalized images which have to be compared (the two filenames are separated by the tabulation character). An illustration of this field is shown in figure 6.

```
Match :0000_000_Norm.bmp   0000_010_Norm.bmp
Match :0000_000_Norm.bmp   0000_011_Norm.bmp
Match :0000_000_Norm.bmp   0010_010_Norm.bmp
Match :0000_000_Norm.bmp   0011_010_Norm.bmp
...
```

Fig. 6 – Illustration of the field `Match` required by the modules *features extraction* and *matching*

Options :

1. `-n` : segmentation is not performed. The input images have already been normalized.
2. `-c` : facultative option used to save the binary iris code files. From an input normalized image called `image_Norm.bmp`, the binary code file will be saved as `image_Norm.cod`.

Path_results : path to the directory which contains the output files, i.e. the text results file (called `Result_file`) and the binary code files (if you have mentioned the standard option `-c`).

Result_file : this file (which is located at `Path_results`) contains on each line the score (Hamming distance) of a comparison between two iris codes. A sample of a such file is given in figure 7.

```
0000_000 0000_010 0.232323
0000_000 0000_011 0.246633
0000_000 0010_010 0.419192
0000_000 0011_010 0.463805
...
```

FIG. 7 – A sample of the results file `Result_file` produced by the module *matching* (name of the target iris on the first column, name of the test iris on the second and score on the third)

IV.Experiments

To test the OSIRIS reference system, two experiments have been carried out in accordance with the protocol defined in part II.2. In the first experiment, we only use the data acquired with the OKI device whereas in the second experiment, we only use those acquired with the CASIA device.
We remind also that the segmentation step is optimized for the Casia v1 database giving poor results when applied to other database. Some examples of mistakes which can appear at this step (using the CBS database) are shown in figure 8. In order to have results which are representative of the whole method, we will provide results obtained with the original eye images from the CBS database (and so segmented by the module *segmentation*) and results obtained with manually segmented images.

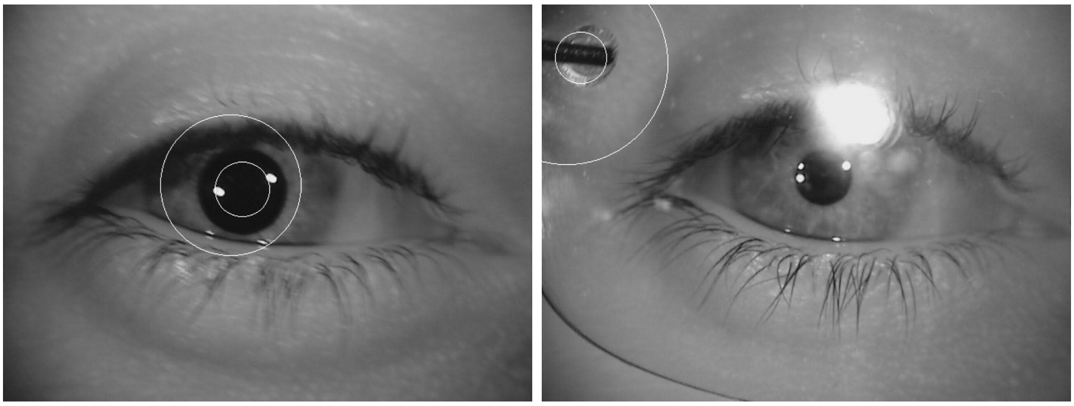


FIG. 8 – Some mistakes which can appear at the segmentation step (these images were aquired using the OKI device)

1.Experiment OKI

The experiment *OKI* consists in using only the iris images acquired with the OKI device. Results are presented below in tab.(2) and tab.(3) :

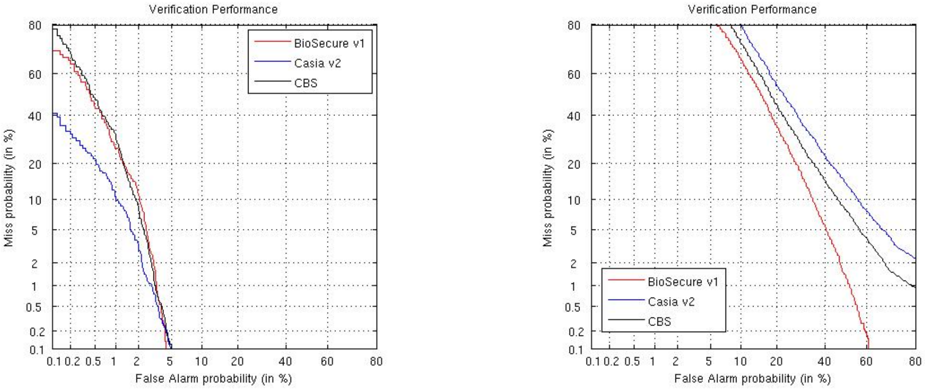
databases	EER
BioSecure v1	2.85%
Casia v2	2.18%
CBS	2.73%

Tab. 2 – Performances measures obtained with manually segmented images

databases	EER
BioSecure v1	24.5%
Casia v2	32.3%
CBS	28.4%

Tab. 3 – Performances measures obtained with original iris images

The corresponding DET curves are displayed in figure 9.



[a] [b]

Fig. 9 – DET curves of reference system performance scores using [a] the manually segmented images and [b] the original iris images

2.Experiment PATTEK

The experiment *PATTEK* consists in using only the iris images acquired with the PATTEK device. Results are presented below in tab.(4) and tab.(5) :

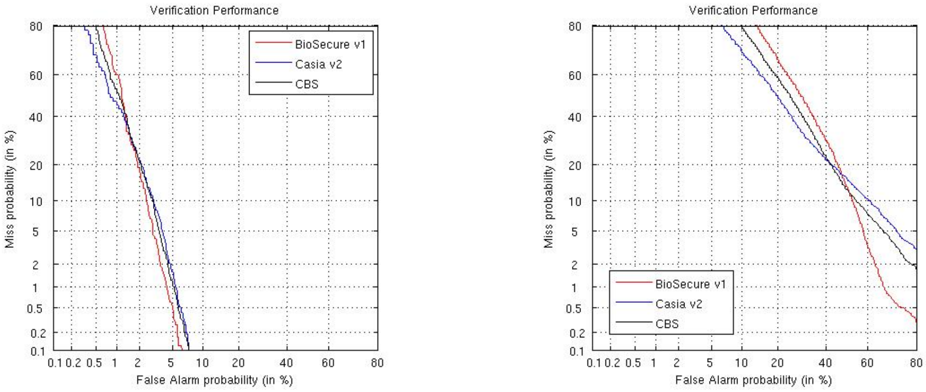
databases	EER
BioSecure v1	3.35%
Casia v2	4.07%
CBS	3.75%

Tab. 4 – Performances measures obtained with manually segmented images

databases	EER
BioSecure v1	35.7%
Casia v2	30.7%
CBS	33.1%

Tab. 5 – Performances measures obtained with original iris images

The corresponding DET curves are displayed in figure 10.



[a] [b]

Fig. 10 – DET curves of reference system performance scores using [a] the manually segmented images and [b] the original iris images

Conclusion

The OSIRIS reference system has been described and tested on the CBS database. This reference system is made of several modules but it is not completely modular. Indeed, the two modules *feature extraction* and *matching* can currently not be dissociated. Furthermore, the module *segmentation* is only suited to the Casia v1 database. That's why the results provided by the system are not satisfactory when the module *segmentation* is applied to other databases than Casia v1.

Acknowledgments

I express here my sincere gratitude to Anouar Mellakh and Emine Krichen for their availability, explanations and kindness.

Reference

[1] J.G. Daugman, High confidence visual recognition of persons by a test of statistical independence, *IEEE Trans. Patt. Anal. Mach. Intell.*, 15(11) :1148–1161, 1993.

ANNEX B

GUIDELINES TO RUN A BENCHMARKING EXPERIMENT FOR FINGERPRINT

A biometric reference system for fingerprint

NIST Fingerprint Image Software 2

`howto last updated - 23/04/2007`

Abstract. *The NIST reference system uses a minutiae based matcher to authenticate a person's identity. Firstly, the minutiae detection algorithm relies on binarization of each grayscale input image in order to locate all minutia points (ridge ending and bifurcation). Then, the matching algorithm computes a match score between the minutia pairs from any two fingerprints using the location and orientation of the minutiae points. The matching algorithm is rotation and translation invariant. Finally, we consider that two fingerprints are from the same finger when the match score is higher than a threshold.*

Introduction

The NIST reference system is made of replaceable modules and the source code is written in C. This document describes this system and explains how to use it in practical to make an automatic verification experiment based on fingerprint modality. This document is divided into four parts. In the first part, we present how to install the system. Then, in the second part, the database used and the protocol which have been defined are described. In the third part, we move on to the explanation of each module and finally, the experimental results obtained in authentication are shown in the last part.

I.Installation

The full system has been tested under Linux (Red Hat 3.3.3-7). For the compilation step, we have used gcc 3.3.3.

1.NFIS2

The reference system is entirely based on the use of the NIST Fingerprint Image Sofware 2 (NFIS2). NFIS2 contains software technology developed for the Federal Bureau of Investigation (FBI) and it is designed to facilitate and support the automated manipulation and processing of fingerprint images. Source code for over 50 different utilities and an extensive user's guide are distributed on CD-ROM free of charge. This CD-ROM is subject to U.S. export control laws.

To order this CD-ROM, you have to follow the instructions given on the next web page :

http ://fingerprint.nist.gov/NFIS/index.html

2.Source code compliation and installation

To compile and install the NFIS2, you have to follow the next steps :

1. copy the top-level file *makefile.mak* and the folders *bin*, *include*, *lib* and *src* from the CD-ROM to your home directory */home*.
2. the header file *include/little.h* needs to be edited prior to compilation. Specifically, the definition for the installation directory must be changed :
   - enter in the include directory
     ```
     >cd include
     ```
   - change the permission on the header file *little.h*
     ```
     >chmod +w little.h
     ```
   - edit the header file *little.h*
     ```
     >emacs little.h
     ```
   - modify the line 8 of the header file
     ```
     #define INSTALL_DIR "/home"
     ```
3. to automatically change permission on all directories and compilation scripts, you have to launch the script *bin/setup.sh* :
   - enter in the bin directory
     ```
     >cd bin
     ```
   - execute the shell script *setup.sh*
     ```
     >sh setup.sh /home
     ```
4. compile and install the source code typing the next commands in your home directory :
   ```
   >make -f makefile.mak PROJDIR=/home depend
   >make -f makefile.mak PROJDIR=/home install
   ```

A succesful compilation will produce 57 executables files stored in the *home/bin* directory.

II.Database and protocol

1.MCYT-100 database

MCYT-100 is a single session database which contains the 10 fingerprints of 100 individuals.
With the aim of evaluating the automatic recognition systems under different acquisition conditions, the database includes 12 different samples of each fingerprint considering three different levels of control[1] :

- 3 samples with low level of control
- 3 samples with medium level of control
- 6 samples with high level of control

---

[1] the level of control depends on the position restrictions of the finger on the screen sensor.

Furthermore, each sample is acquired using two different sensors :

- a CMOS-based capacitive capture device, model 100SC from Precise Biometrics, producing a 500 dpi, 300*300 pixel image
- an optical scanning device, model UareU from Digital Persona, producing a 500 dpi, 256*400 pixel image

Therefore, in each capture session each individual provides a total number of 240 fingerprint images to the database (10 prints × 12 samples/print × 2 sensors).

To have a full description of the MCYT database, you can download the next documentation :

http ://atvs.ii.uam.es/doc/MCYT_DB_paper.pdf

2.Protocol

The data acquired with the two different sensors will not be mixed. So, two experiments will be carried out :

- an experiment with data acquired using the optical sensor
- an experiment with data acquired using the capacitive sensor

For each experiment, data consist of 12000 fingerprint images from the 10 fingers of 100 contibutors. We consider the different fingers as different users enrolled in the system, thus resulting in 1000 users with 12 impressions per user.

For the both experiments, the protocol defined is the same. We use one impression per finger with low control during the acquisition as a template. In genuine trials, the template is compared to the other 11 impressions available (2 with low control, 3 with medium control and 6 with high control). The impostor trials are obtained by comparing the template to one impression with high control of the same finger of all the other contributors. The total number of genuine and impostor trials are therefore $1000 \times 11 = 11000$ and $1000 \times 99 = 99000$, respectively.

N.B : For the following, we suppose that the MCYT-100 database has been installed at */home/mcyt* location.

III.Presentation of the method

Among all the utilities and packages provided by the NIST Fingerprint Image Software 2, only two packages are needed for our evaluation and tests :

- MINDTCT for minutiae extraction
- BOZORTH3 for fingerprint matching

1.MINDTCT

a.Description

The MINDTCT module automatically locates and records minutiae (ridge ending and bifurcation) in a fingerprint image whereas false minutiae (islands, lakes, holes. . .) are removed. The module includes a quality measure which is associated with each detected minutia point. Indeed, even after performing the removal step, false minutiae potentially remain in the list of minutiae. So a robust quality measure can help to manage this.

In practice, the module makes the following steps (each step corresponds to one module, which permits other alternative approaches to be implemented and substituted into the process) :

- generation of image quality map
- binarization
- minutiae detection
- removal of false minutiae
- counting of ridges between a minutia point and its nearest neighbors
- minutiae quality assessment

To have a more comprehensive description of the MINDTCT module, you can consult the NFIS2 reference manual, from page 56 to 77.

b.Software presentation

To detect minutiae from a fingerprint image, we use the executable `mindtct` which is located at */home/bin*. The executable usage is given in the following :

```
>./mindtct -b inputfile outputroot
```

The parameters of the mindtct executable are the fingerprint filename to be processed (*inputfile*) and the root name of the output files (*outputrooot*). The option `-b` performs image enhancement on low contrast images (this option affects only low contrast images, others are unchanged).

c.Input illustration

The MINDTCT module takes as input a fingerprint image. For our evaluation, this image is from the MCYT-100 database. Some example images of this database are shown in figure (1).

N.B : The MINDTCT module does not support BMP file format. So all images of the MCYT-100 database have beforehand to be converted in JPEG format. To do that, we use the following command :

```
>./convert inputimage.bmp inputimage.jpg
```

FIG. 1 – Examples of acquired images using the optical capture device (on the left) and the capacitive capture device (on the right)

d. Output illustration

The output of the MINDTCT module is a text file named *outputrooot.xyt*. This file reports the minutiae detection results. It has one space delimited line per minutiae containing its x and y coordinate (the pixel origin is at the bottom left of the image), direction angle theta[2] (directions are pointing out and away from ridge ending and bifurcation) and the minutia quality (value on the range of 1 to 99). A sample of a such text file is shown in figure (2).

```
23 178 247 11
24 165 236 11
30 141 214 11
51 214 247 94
52 151 225 46
56 240 45  77
59 215 56  87
72 242 214 85
...
```

FIG. 2 – A sample of the fingerprint minutiae file (x coordinate on the first column, y coordinate on the second, direction angle on the third and quality measure on the fourth)

<u>N.B</u> : The mindtct executable creates a lot of other text output files than *outputrooot.xyt*. But here, we have only described the text file which will be used as input by the next module (BOZORTH3). To have a comprehensive description of the other text output files, you can consult the NFIS2 reference manual, from page 166 to 168.

---

[2] direction angle theta is in degree.

2.BOZORTH3

a.Description

The BOZORTH3 module computes a match score between the pairs of minutiae from any two fingerprints. Three key things are important to note regarding this fingerprint matcher :

- it is only based on the location and orientation of the minutiae points
- only the 150 highest-quality minitiae are used
- the algorithm is designed to be rotation and translation invariant

The algorithm is comprised of three major steps :

1. Construct intra-fingerprint minutia comparison tables :
   The first step in the bozorth3 matcher is to compute relative measurements from each minutia in a fingerprint to all other minutia in the same fingerprint. These invariant measurements are distance between two minutiae and angle between each minutia's orientation and the intervening line between both minutiae. These measurements are then stored in a minutia comparison table (for the matching step, there will be one table for the gallery fingerprint and one table for each probe (test) fingerprint to be matched against).
2. Construct an inter-fingerprint compatibility table :
   The second step in the bozorth3 matcher is to take the intra-fingerprint minutia comparison tables from the two fingerprints (probe and gallery) and look for compatible entries between the two tables. Table entries are considered to be compatible if the corresponding distances and the relative minutiae angles are within a specified tolerance. An inter-fingerprint compatibility table is then generated, only including entries that are compatible.
3. Traverse the inter-fingerprint compatibility table :
   The final step of the matcher consists in traversing and linking table entries into clusters, combining compatible clusters and accumulating a match score. The larger the number of linked compatibility associations, the higher the match score, and the more likely the two fingerprints are from the same person.

To have a more comprehensive description of the BOZORTH3 module, you can consult the NFIS2 reference manual, from page 80 to 86.

b.Software presentation

To compute match scores between fingerprints, we use the executable `bozorth3` which is located at */home/bin*. The executable usage is given in the following :

```
>./bozorth3 -A outfmt=spg -o outfile.log -g galleryfile.xyt -P probe.lst
```

The match scores between the gallery fingerprint minutiae file (*galleryfile.xyt*) and all probe (test) fingerprint minutiae files (whose names are contained in a single list file *probe.lst*) are computed and stored in an output log file (*outputfile.log*). The option `-A outfmt=spg` mentions that the output file contains on each line score (s), probe (p) and gallery (g) filename.

c. *Input illustration*

The input minutiae files are expected to be in xyt-format, a simple text file format that is produced by the minutiae detector program `mindtct` (see fig.(2)).

d. *Output illustration*

The output of the BOZORTH3 module is a log file named *outputfile.log.* It has one space delimited line per matching test containing the score, the probe and the gallery fingerprint minutiae filename. A sample of a such text file is shown in figure (3).

```
206 /home/mindtct/0000/dp_0000_0_1.xyt /home/mindtct/0000/dp_0000_0_0.xyt
260 /home/mindtct/0000/dp_0000_0_2.xyt /home/mindtct/0000/dp_0000_0_0.xyt
162 /home/mindtct/0000/dp_0000_0_3.xyt /home/mindtct/0000/dp_0000_0_0.xyt
226 /home/mindtct/0000/dp_0000_0_4.xyt /home/mindtct/0000/dp_0000_0_0.xyt
7   /home/mindtct/0001/dp_0001_0_11.xyt /home/mindtct/0000/dp_0000_0_0.xyt
5   /home/mindtct/0002/dp_0002_0_11.xyt /home/mindtct/0000/dp_0000_0_0.xyt
13  /home/mindtct/0003/dp_0003_0_11.xyt /home/mindtct/0000/dp_0000_0_0.xyt
6   /home/mindtct/0004/dp_0004_0_11.xyt /home/mindtct/0000/dp_0000_0_0.xyt
...
```

FIG. 3 – A sample of the score file (score on the first column, probe fingerprint minutiae filename on the second and gallery fingerprint minutiae filename on the third)

To have a more comprehensive description of the bozorth3 executable, you can consult the NFIS2 reference manual, from page 113 to 115.

IV. *Experiments*

To test the NIST reference system, two experiments have been carried out in accordance with the protocol defined in part II.2. In the first experiment, we only use the fingerprints acquired with the optical sensor whereas in the second experiment, we only use those acquired with the capacitive sensor. We remind here that for each experiment, the total number of genuine and impostor trials are 11000 and 99000 respectively.

1. *Experiment 1*

The experiment 1 consists in using only the fingerprints acquired with the optical scanning device (model UareU from Digital Persona). The DET curve obtained in accordance with the reference protocol is displayed in figure (4). The EER is equal to 3.22%.

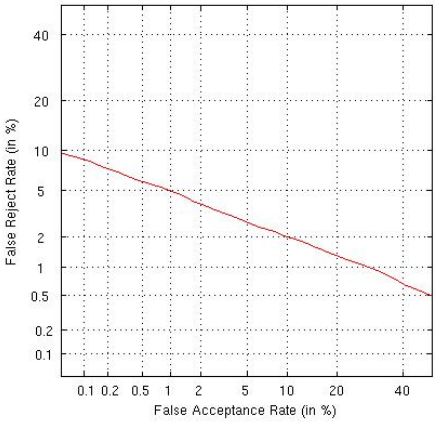


Fig. 4 – DET curve obtained using the fingerprints acquired with the optical sensor

2.Experiment 2

The experiment 2 consists in using only the fingerprints acquired with the CMOS-based capacitive capture device (model 100SC from Precise Biometrics). The DET curve obtained in accordance with the reference protocol is displayed in figure (5). The EER is equal to 8.27%.

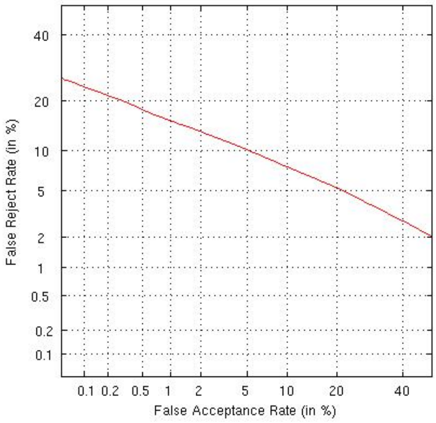


Fig. 5 – DET curve obtained using the fingerprints acquired with the capacitive sensor

Conclusion

The NIST reference system has been described and tested on the MCYT-100 database. This reference system is made of replaceable modules which is of great interest. From now, a researcher could indeed show the improvement of his own new module simply by replacing the corresponding module in the reference system.