République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique
Département d'ELECTRONIQUE
Année scolaire : 2009 - 2010

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Projet de fin d'études
en vue de l'obtention du diplôme
d'ingénieur d'état en ELECTRONIQUE

Thème

Implémentation temps réel sur FPGA d'une combinaison séquentielle de l'algorithme PCA avec un réseau de neurones multicouche pour la reconnaissance de visages

Proposé et dirigé par :

Pr. L. HAMAMI

Réalisé par :

M. KHARCHI Zakaria Oussama
M. NAIT MEZIANE Mohamed

Promotion : Juin 2010
Ecole nationale polytechnique, 10, Avenue Hassen Badi, BP 182 El-Harrach, Alger, Algérie

Dédicace

A ma mère, ma mère, ma mère et mon père
A mon frère
A toute ma famille
A ma promotrice : L.HAMAMI
A mon binôme : Mohamed
A tous mes amis
Je dédie ce modeste travail

Zakaria Oussama

A mes parents
A mes frères et soeurs
A toute ma famille sans exception
A mon collègue : Zaki
A tous mes amis
Je dédie ce modeste travail

Mohamed

ملخص :

يتمثل هذا العمل في إنجاز نظام من أجل التعرف على الأوجه يستند على الجمع التعاقبي بين الخوارزمية PCA مع شبكة عصبية اصطناعية متعددة الطبقات.
التصديق على عمل النظام تم بواسطة البرنامج MATLAB. قاعدة البيانات المستخدمة تتكون من 2410 صورة ثابتة لأوجه في حالة أمامية.
من أجل تشكيل النظام على دارة FPGA، استعملنا لغة البرمجة المادية VHDL.
كلمات مفتاحية : التعرف على الأوجه، PCA، شبكة عصبية متعددة الطبقات، FPGA.

Résumé :

Ce travail consiste en la réalisation d'un système pour la reconnaissance de visages basé sur une combinaison séquentielle de l'algorithme PCA avec un réseau de neurones artificiel multicouche.

La validation du fonctionnement du système a été faite par le logiciel Matlab. La base de données utilisée est composée de 2410 images fixes de visages en position frontale.

Dans le but de faire une implémentation sur un circuit FPGA, nous avons utilisé le langage de programmation matérielle VHDL.

Mots-clés : reconnaissance de visages, PCA, réseau de neurones multicouche, FPGA.

Abstract :

This work consists of the realization of a system for face recognition based on a sequential combination of the PCA algorithm with a multi-layer artificial neural network.

The validation of system functioning was made by the Matlab software. The database used is made up of 2410 still images of faces in frontal position.

In order to make an implementation on an FPGA circuit, we used the hardware description language VHDL.

Keywords : Face recognition, PCA, Multi-layer neural network, FPGA.

Remerciements

Nous adressons nos plus vifs remerciements à notre promotrice Pr. L. HAMAMI pour les précieux conseils qu'elle nous a donnés, pour les connaissances qu'elle nous a octroyé et surtout pour son assistance malgré ses nombreuses occupations.

Nous remercions de même monsieur M. HADDADI, professeur à l'école nationale polytechnique, qui nous a fait l'honneur de présider le jury.

Nos remerciements vont aussi à M^{lle} A. MOUSSAOUI pour avoir accepté d'examiner ce modeste travail.

Nous remercions M. S. MAKHLOUFIA pour son aide précieuse.

Nous tenons à remercier également l'ensemble des enseignants qui ont contribué à notre formation.

Nous voudrions enfin remercier tous nos collègues de la promotion 2010 et en particulier ceux de la spécialité électronique.

Sommaire

Table des figures

Liste des tableaux

Introduction générale

La biométrie est un secteur connaissant un succès croissant au fur et à mesure que les techniques d'identification par la biométrie évoluent. Actuellement, les techniques de reconnaissance évoluent rapidement. Le nombre de produits utilisant des technologies issues de la biométrie seront de plus en plus nombreux. Et présent parmi nous.

La reconnaissance de visages basée sur le traitement des images 2D s'est bien développée au cours de ces dernières années, et plusieurs techniques ont été proposées. Malgré l'avancement de la recherche réalisée durant ces trois dernières décennies, la reconnaissance robuste de visages reste très difficile. Les méthodes actuelles sont efficaces lorsque les conditions de prise de vue sont très bien contrôlées. En effet, la variation de luminosité ou le changement de prise de vue causent des problèmes sérieux pour de nombreux systèmes de reconnaissance existants.

L'une des méthodes les plus utilisées est la PCA (Principal Components Analysis), basée sur un algorithme qui identifie des motifs dans des données et qui exprime les données de façon à accentuer leurs similitudes et différences.

L'objectif de notre travail est de combiner l'algorithme PCA avec un réseau de neurones multicouche pour construire un système pour la reconnaissance de visages. Ce système devra travailler en temps réel, donc son implémentation sur circuit dédié en l'occurrence un FPGA a été réalisée.

Notre mémoire est organisé de la manière suivante :

Dans le premier chapitre, nous donnerons quelques notions fondamentales sur le traitement d'images et la biométrie.

Le deuxième chapitre sera consacré à une étude comparative de différentes techniques utilisées pour la reconnaissance de personnes. Nous donnerons aussi les critères essentiels qui justifient le choix de la reconnaissance de visages.

Le troisième chapitre fera l'objet d'une étude théorique de l'algorithme PCA et des réseaux de neurones multicouches.

Dans le quatrième chapitre nous présenterons la simulation faite sur Matlab du système de reconnaissance réalisé.

Le dernier chapitre contiendra la description de l'implémentation de notre système sur un circuit FPGA. Cette implémentation sera faite avec le langage VHDL.

Chapitre 1

Généralités sur le traitement d'images et la biométrie

1.1 Le traitement d'images

1.1.1 Définition

Le traitement d'images est l'ensemble des méthodes qui permettent de rendre une image plus intelligible pour l'être humain ou plus facile à manipuler par la machine, en mettant en exergue certaines de ses caractéristiques, en vue d'une application donnée.

C'est une technique qui fait appel à plusieurs disciplines : La théorie du signal, la théorie des systèmes, l'analyse numérique, les statistiques, la théorie de l'information, la neurophysiologie et la psychophysique, l'optique, l'électronique et l'informatique pour ne citer que les principales (figure 1.1).

Les différentes relations avec des domaines variés engendrent une grande diversité dans les traitements que l'on peut effectuer sur des images. Pour illustrer cette diversité, on peut citer quelques exemples :

– Rendre nette une image floue ;

- Améliorer le contraste d'une image ;
- Filtrer des parasites sur une image ;
- Comprimer le nombre d'échantillons d'une image numérique par un facteur de 100 ou plus et d'une séquence d'images (TV numérique) par un facteur 1000 ou plus ;
- Reconnaître et compter des bactéries dans des préparations microscopiques ;
- Reconnaître des visages, empreintes digitales et des signatures ;
- Aider les pilotes et les conducteurs à s'entraîner ;
- Diriger un véhicule sans chauffeur ;
- Surveiller les machines-outils ;
- Aider les robots mécaniques à voir ;
- Etc.

FIGURE 1.1 – Relations du traitement d'images avec d'autres disciplines

Il faut noter aussi que la principale difficulté dans la réalisation d'une application quelconque de traitement d'images est l'absence d'un traitement idéal universel qui résoudrait plusieurs problèmes à la fois. Ainsi, chaque application particulière nécessite une étude en soi. [1]

Les algorithmes de traitement d'images impliquent des volumes de calcul assez importants. L'essor de l'implémentation de ces algorithmes est fortement lié à l'avènement sur le marché des processeurs de traitement du signal (DSP, acronyme de Digital Signal Processor) et des circuits FPGA (Field Programmable Gate Array).

1.1.2 Notions fondamentales

Définition d'une image

Une image est une représentation numérique bidimensionnelle d'une scène analogique réelle située en général dans un espace tridimensionnel. Elle possède l'information pour chaque point de son intensité lumineuse fournie par les capteurs (appareils photo, cameras, scanner, ...).

On peut la décrire par une fonction $f(x, y)$ où :

f	:	est la fonction d'intensité lumineuse définie dans un domaine borné ;
x, y	:	coordonnées cartésiennes d'un point de l'image ;
$f(x, y)$	:	niveau de gris en ce point. [2]

Pixel

Un pixel (contraction de « picture element ») est l'unité de base constituant l'image, si le bit est la plus petite unité d'information que peut traiter un ordinateur, le pixel est le plus petit élément que peuvent manipuler les matériels et logiciels d'affichage ou d'impression, il est caractérisé par sa position (i en abscisse et j en ordonné) et sa valeur (niveau de gris ou couleur). [2]

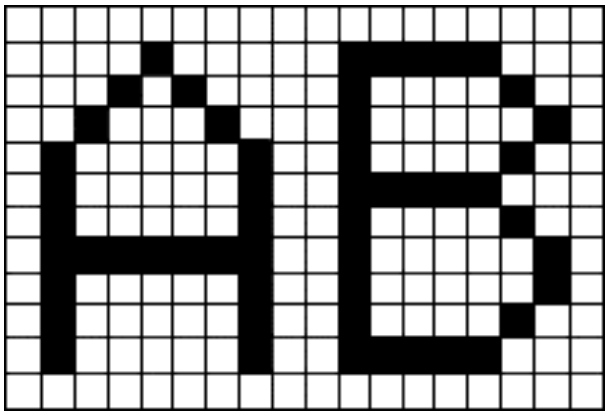


FIGURE 1.2 – Les lettres A et B écrites avec un groupement de pixels

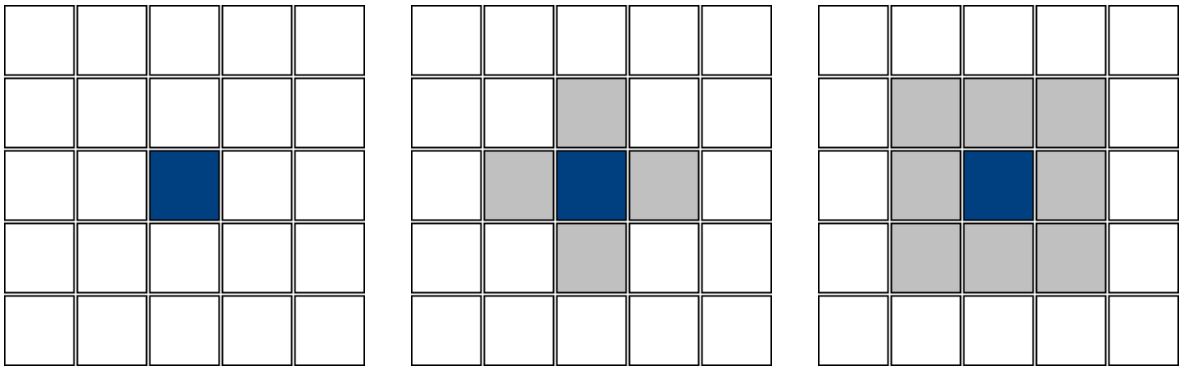
Voisinage d'un pixel

Le voisinage d'un pixel est composé de tous les pixels qui l'entourent immédiatement. Dans une image numérique, on distingue deux types de connexités relatives au voisinage : la 4-connexités et la 8-connexités (figure 1.3).

Connexité

La connexité entre les pixels d'une image est un concept très important utilisé surtout dans l'établissement des frontières des objets et l'identification des composants d'une région dans une image.

Pour établir si deux pixels sont connectés, nous devons déterminer s'ils sont adjacents quelques parts (par exemple, s'ils appartiennent à un 4-voisinage) et si leurs niveaux de gris respectent un certain critère de similarité. [2]



(a) Un pixel (b) La 4-connexités (c) La 8-connexités

FIGURE 1.3 – Voisinage d'un pixel

Contraste

C'est l'opposition marquée entre deux régions juxtaposées d'une image, plus précisément les régions sombres et les régions claires de cette image [3]. Il est défini en fonction des luminances L_i et L_{i+1} de deux zones de l'image par l'équation :

$$C = \frac{L_i - L_{i+1}}{L_i + L_{i+1}}$$

Résolution d'une image

La résolution d'une image est définie par un nombre de pixels par unité de longueur de la structure à numériser (classiquement en dpi (dots per inches) ou ppp (points par pouce)).

Ce paramètre est défini lors de la numérisation et dépend principalement des caractéristiques du matériel utilisé lors du processus de numérisation. Plus le nombre de pixels est élevé par unité de longueur de la structure à numériser, plus la quantité d'information qui décrit cette structure est importante et plus la résolution est élevée. [3]

Image en niveaux de gris

Dans une image en niveaux de gris, la couleur d'un pixel peut prendre des valeurs allant du noir au blanc, en passant par un nombre fini de niveaux intermédiaires.

En général, les images en niveaux de gris renferment 256 teintes de gris. Par convention la valeur zéro représente le noir (intensité lumineuse nulle) et la valeur 255 le blanc (intensité lumineuse maximale). Chaque pixel n'est donc plus représenté par un bit, mais par un octet. Le nombre de niveaux de gris dépend du nombre de bits utilisés pour décrire la « couleur »de chaque pixel de l'image. Plus ce nombre est important, plus les niveaux possibles sont nombreux. [4]

FIGURE 1.4 – Valeurs des niveaux de gris et teintes correspondantes

Image binaire

Ces images sont constituées de pixels qui ne peuvent avoir que deux états : noir ou blanc. On les appelle des images « au trait ». On trouve parmi elles des pages de texte, des signatures, des plans, des dessins, etc. [5]

Image en couleurs

Une image en couleurs est censée représenter le mieux possible la réalité. La représentation des couleurs s'effectue de la même manière que les images monochromes avec cependant quelques particularités.

En effet, il faut tout d'abord choisir un modèle de représentation. On peut représenter les couleurs à l'aide de leurs composantes primaires. Les systèmes émettant de la lumière (écrans d'ordinateurs, ...) sont basés sur le principe de la synthèse additive : les couleurs sont composées d'un mélange de rouge, vert et bleu (modèle R.V.B.). [4]

1. *Représentation en couleurs réelles (Images en "vraies couleurs" (ou 24 bits))* :
   Elle consiste à utiliser 24 bits pour chaque point de l'image (figure 1.5). Huit bits sont employés pour décrire la composante rouge (R), huit pour le vert (V) et huit pour le bleu (B), ce qui fait que les images en vraies couleurs sont des images très « lourdes ». Il est ainsi possible de représenter environ 16,7 millions de couleurs différentes simultanément. Par ailleurs l'œil humain n'est pas capable de distinguer autant de couleurs.

Valeur R	Valeur V	Valeur B	Couleur correspondante	Commentaire
0	0	0		Noir
0	0	1		Un peu moins noir (nuance impossible à discerner du noir par l'œil humain)
...	...	...	...	...
0	0	255		Bleu
...	...	...	...	...
0	255	0		Vert
...	...	...	...	...
255	0	0		Rouge
...	...	...	...	...
128	128	128		Couleur intermédiaire (gris)
255	255	255		blanc

FIGURE 1.5 – Représentation en vraies couleurs (24 bits)

2. *Représentation en couleurs indexées* :
   Afin de diminuer la charge de travail nécessaire pour manipuler des images en 24 bits, on peut utiliser le mode de représentation en couleurs indexées. Le principe consiste à déterminer le nombre de couleurs différentes utilisées dans l'image, puis à créer une table de ces couleurs en attribuant à chacune une valeur numérique correspondant à sa position dans la table.
   La table, appelée palette, comporte également la description de chacune des couleurs, sur 24 bits (figure 1.6).
3. *Autres modèles de représentation* :
   Le modèle R.V.B. représentant toutes les couleurs par l'addition de trois composantes fondamentales, n'est pas le seul possible. Il en existe de nombreux autres. L'un d'eux est particulièrement important. Il consiste à séparer les informations de couleurs (chrominance) et les informations d'intensité lumineuse (luminance).

Palette avec les codes RVB				Couleur correspondante
Couleur	Code RVB			
0	255	255	255	
1	255	255	204	
2	255	255	153	
3	255	255	102	
4	255	255	51	
5	255	255	0	
…	…			…

FIGURE 1.6 – Exemple d'une palette de couleurs

Il s'agit du principe employé pour les enregistrements vidéo. La chrominance est représentée par deux valeurs (selon des modèles divers) et la luminance par une valeur.

Bruit dans l'image

Le bruit dans une image est défini comme étant un phénomène de brusque variation d'un pixel isolé par rapport à ses voisins, il affecte la qualité de l'image et il provient soit du dispositif d'acquisition (scanner, caméra, amplification, quantification, ...) soit de la scène elle-même (poussières, rayures, ...). [5]

Histogramme

L'histogramme des niveaux de gris ou des couleurs d'une image est une fonction qui donne la fréquence d'apparition de chaque niveau de gris (couleur) dans l'image.

Pour diminuer l'erreur de quantification, pour comparer deux images obtenues sous des éclairages différents ou encore pour mesurer certaines propriétés de l'image, on modifie souvent l'histogramme correspondant.

L'histogramme permet de donner un grand nombre d'informations sur la distribution des niveaux de gris (couleur) et de voir entre quelles bornes est repartie la majorité des niveaux de gris (couleur) dans le cas d'une image trop claire ou d'une image trop foncée (figure 1.7).

L'histogramme peut être utilisé pour améliorer la qualité d'une image (Rehaussement d'image) en introduisant quelques modifications, pour pouvoir extraire les informations utiles de celle-ci. [4]

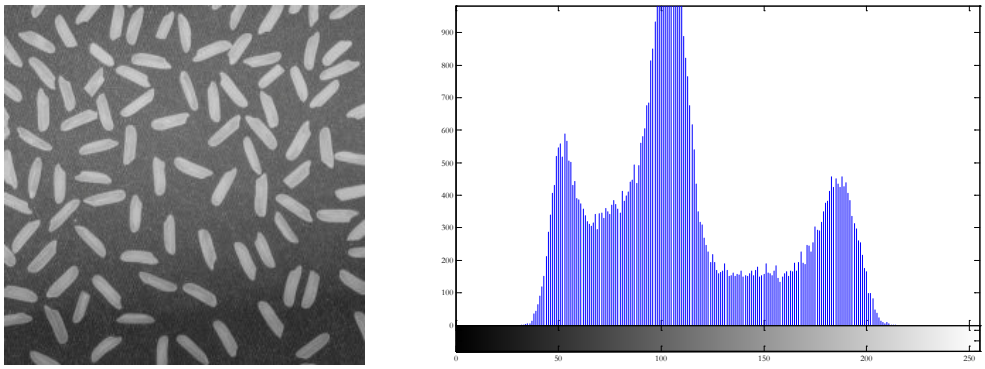


FIGURE 1.7 – Une image et son histogramme

1.1.3 Etapes d'analyse des images

La plupart des applications du traitement d'images passent par les étapes suivantes :

- **Acquisition** : Echantillonnage, Quantification.
- **Analyse globale de l'image et transformations ponctuelles** : Histogramme, statistiques (moyenne, écart-type, etc.), transcodage (palette de couleur [LUT]) et classification.
- **Opérations entre images** : Indices, ratio, différence, opérations logiques, masques et seuillage.
- **Amélioration, filtrage et segmentation** : Opérations de convolution (lissage, rehaussement, détection de contours) ; squelettisation, vectorisation.
- **Interprétation et sémantique** : Cartographie thématique (classification automatique et supervisée), cartographie vecteur, représentation des graphes et de la topologie.

1.1.4 Quelques techniques du traitement d'images

Avant d'extraire les objets et d'analyser une image, il est souvent nécessaire d'améliorer sa qualité en lui appliquant quelques traitements (modification d'histogramme, binarisation, amélioration du contraste, élimination de bruits, etc.).

Nous allons présenter dans ce qui suit quelques techniques de base.

Modification d'histogramme

Certaines images sont initialement trop claires, foncées ou bien peu contrastées, cela est dû respectivement au fait que les niveaux de gris de l'image sont tassés vers le haut de l'échelle, vers le bas ou bien sont regroupés dans un intervalle étroit. Ces défauts sont très visibles sur l'histogramme. Le but est de redistribuer les niveaux de gris de l'image afin de leur faire occuper toute la bande de nuances possibles. Le principe est donné sur la figure 1.8.

Cette méthode est basée sur les transformations ponctuelles d'intensité. C'est-à-dire, à chaque pixel d'intensité I_s on associe une intensité $I'_s = T(I_s)$. [2]

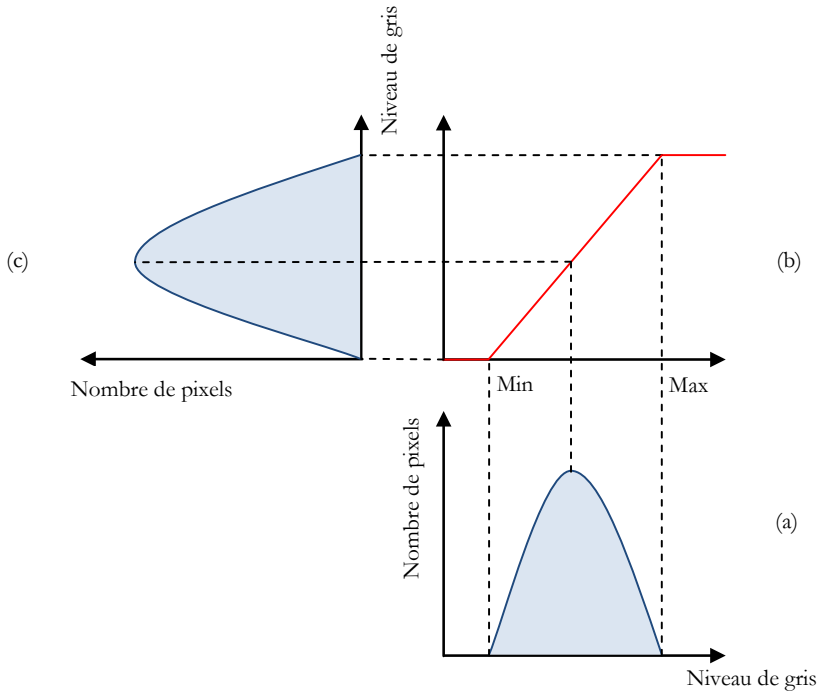


FIGURE 1.8 – Principe de la modification d'histogramme
(a) Histogramme de l'image originale
(b) Fonction de transformation
(c) Histogramme recadré

Binarisation par seuillage

Cette technique consiste à réduire la dynamique d'une image (plusieurs niveaux de gris) à deux niveaux (noir et blanc). Ceci revient à séparer les pixels de l'image en deux classes, la première ayant un niveau maximal (typiquement 255) et la seconde un niveau minimal (0).

La méthode consiste à calculer un seuil d'après l'histogramme de l'image, ce seuil doit être le plus adéquat possible pour ne pas altérer les informations pertinentes de l'image (figure 1.9). [2]

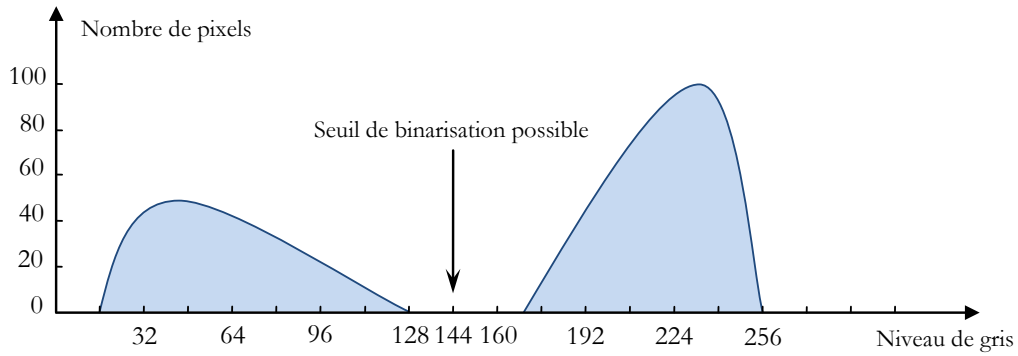


FIGURE 1.9 – Choix du seuil de binarisation sur l'histogramme

Filtrage

Le but du filtrage est essentiellement l'atténuation du bruit et l'accentuation des discontinuités. On peut citer , entre autre , les classes de filtres suivants :

- Les filtres linéaires ;
- Les filtres non linéaires ;
- Les filtres morphologiques.

1. **Les filtres linéaires** :
   Ce type de filtrage est caractérisé par une opération de convolution de sorte que la transformation de chaque pixel est le fruit d'une combinaison linéaire des pixels voisins.

$$I'(i,j) = I(i,j) * filtre(i,j)$$
$$I'(i,j) = \sum_u \sum_v I(i-u, j-v).filtre(i,j)$$

   - *Le filtre moyenneur* :
     Ce filtre considère chaque pixel de l'image et fait la moyenne avec les pixels voisins en utilisant un masque. On obtient une image adoucie en réduisant les variations brusques de niveaux de gris.
     La taille du masque est un paramètre variable. Plus le masque sera de grande dimension, plus l'effet du filtrage est apparent.
     On utilise le masque suivant :

$$H = \frac{1}{N} \begin{vmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{vmatrix}$$

     Où : N est un coefficient de normalisation = somme des coefficients non nuls.
     Chaque pixel est multiplié par le coefficient correspondant, les masques les plus courants sont :

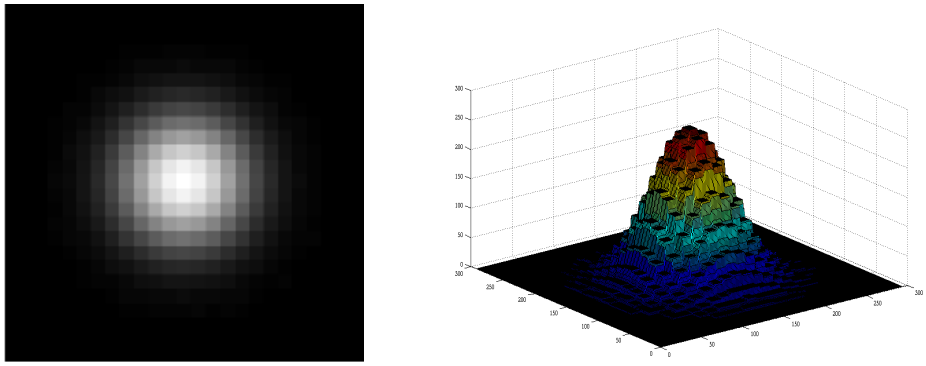
$$H_1 = \frac{1}{9} \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}, \quad H_2 = \frac{1}{10} \begin{vmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{vmatrix}, \quad H_3 = \frac{1}{16} \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$$

     On déplace le masque sur toute l'image et le pixel affecté par la transformation sera le pixel central du masque. [6]

   - *Le filtre gaussien* :
     Pour appliquer ce filtre, il suffit de convoluer l'image initiale avec une Gaussienne $G(x, y, \sigma)$ à deux dimensions donnée par la formule :

$$G(x,y,\sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

     Ce filtre (figure 1.10) effectue la moyenne des pixels voisins avec une pondération par des coefficients discrétisés d'une gaussienne.
     Il a l'avantage de régler le degré de filtrage par le paramètre σ. [2]



(a) Image d'une gaussiène (b) Fonction d'une gaussiène en 3D

FIGURE 1.10 – Exemple du filtre gaussien pour les images

2. **Les filtres non linéaires** :
   Ces opérateurs ont été développés pour pallier les insuffisances des filtres linéaires, surtout la mauvaise conservation des contours.
   Les pixels voisins interviennent suivant une loi non linéaire dans la détermination du pixel central. Un des exemples de ces filtres est le filtre médian.

   – *Le filtre médian* : Le niveau de gris du pixel central est remplacé par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel. La taille du masque dépend de la variance du bruit et de la taille des détails significatifs de l'image traitée.
   La figure 1.11 illustre le fonctionnement d'un filtre médian de fenêtre d'analyse 3×3 sur un exemple. Ici le pixel central ayant le niveau de gris (12) aura le niveau de gris (15).

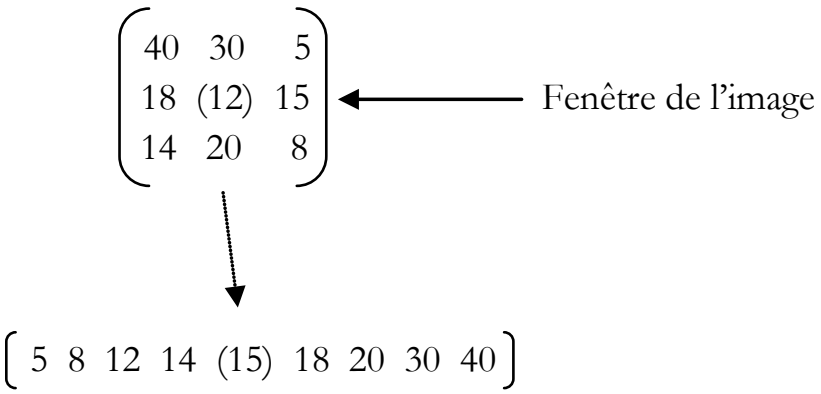


FIGURE 1.11 – Principe de fonctionnement d'un filtre médian

Le filtre médian garde la netteté de l'image pour les éléments de dimensions importantes par rapport au noyau du filtre, mais élimine les détails fins de manière irrémédiable. [2]

3. **Les filtres morphologiques** :
   Les filtres morphologiques sont souvent utilisés pour éliminer des pixels isolés considérés comme un bruit dans une image binarisée. Ces méthodes utilisent un élément structurant.
   Parmi ces opérations morphologiques il y a la dilatation, l'érosion, l'ouverture et la fermeture mathématiques.
   - *La dilatation* : Elle permet d'éliminer les pixels blancs isolés mais ajoute des pixels noirs au contour des objets présents dans l'image. Le résultat de cette opération est l'augmentation de la taille de ces objets.
   - *L'érosion* : Elle permet d'éliminer les pixels noirs isolés au milieu des parties blanches de l'image. Le résultat de cette opération est la diminution de la taille des objets présents dans l'image.
   - *L'ouverture* : L'ouverture est constituée par une opération d'érosion suivie d'une dilatation. Elle permet de retrouver la taille normale des objets de l'image.
   - *La fermeture* : La fermeture est l'opération inverse de l'ouverture, qui consiste à faire subir à l'image une dilatation suivie d'une érosion. Elle permet aussi de retrouver la taille normale des objets de l'image. [5]

Segmentation

La segmentation est l'un des éléments les plus importants dans le traitement d'image car elle permet l'extraction des entités et des objets pour des traitements ultérieurs telle la reconnaissance de formes.

Les algorithmes de segmentation d'image reposent généralement sur l'une des deux propriétés suivantes : la *discontinuité* et la *similarité*. Ceci donne naissance à deux approches couramment qualifiées d'approche « frontières » et d'approche « régions ».

- *Approche frontières* : Le partitionnement de l'image est basé sur les changements brusques des niveaux de gris. Leur centre d'intérêt est alors l'extraction des points isolés et la détection des lignes de contours dans une image.
  Il existe deux approches selon lesquelles se fait la détection de contour :
  - ℘ **Approche gradient** : détermination des extrema locaux dans la direction du gradient.
  - ℘ **Approche Laplacien** : détermination des passages par zéro du Laplacien. Les différentes approches existantes se classent ensuite suivant la manière d'estimer les dérivées de la fonction d'intensité, il existe alors :
    - ◇ Les méthodes dérivatives.
    - ◇ Les méthodes morphologiques.
- *Approche régions* : L'approche duale à la détection de contours pour la segmentation d'image est l'approche par régions. Elle repose sur la recherche de zones possédant des attributs communs, de luminosité, de textures, ...
  Les méthodes de l'approche régions aboutissent directement à une partition de l'image, chaque pixel étant affecté à une région.
  Contrairement aux techniques d'extraction de contours, la segmentation en régions homogènes est basée sur les propriétés intrinsèques des régions. Le choix de ces propriétés détermine ce qu'on appelle « le critère de segmentation ».
  Dans cette approche, on peut citer :
  - ℘ **Les méthodes de classification**.
  - ℘ **Les méthodes Markoviennes**.
  - ℘ **Les méthodes structurales**. [2]

1.2 La biométrie

1.2.1 Présentation de la biométrie

Le contexte

La croissance internationale des communications, tant en volume qu'en diversité (déplacement physique, transaction financière, accès aux services, ...), implique le besoin de s'assurer de l'identité des individus. L'importance des enjeux, motive les fraudeurs à mettre en échec les systèmes de sécurité existants.

La lutte contre les fraudes bancaires continue et les constructeurs de distributeurs automatiques s'engagent sur la voie des nouvelles technologies comme la biométrie ou les technologies sans contact. Or la commercialisation de ces nouvelles technologies tarde à être lancée car l'opinion publique semble plutôt réticente. Cette fraude a coûté à la France 241,6 millions d'euros en 2004. Les taux de fraude sur les transactions internationales sont beaucoup plus élevés que ceux sur les transactions nationales. Les voleurs s'appuient sur les différences de normes entre les pays.

Il y a donc un intérêt grandissant pour les systèmes d'identification et d'authentification. Leur dénominateur commun, est le besoin d'un moyen simple, pratique, fiable, pour vérifier l'identité d'une personne, sans l'assistance d'une autre personne. Le marché du contrôle d'accès s'est ouvert avec la prolifération de systèmes, mais aucun ne se révèle efficace contre la fraude, car tous utilisent un identifiant externe tel que : badge/carte, clé, code, ...

Il est fréquent d'oublier le code d'accès. Pour éviter cet oubli, beaucoup de personnes écrivent ce code sur un carnet, perdant ainsi toute confidentialité. Les moyens biométriques, permettent une authentification sûre, ce qui n'est pas le cas avec les mots de passe ou les cartes (badges). Ces derniers peuvent être utilisés par des tiers non autorisés.

Le niveau de sécurité d'un système est toujours celui du maillon le plus faible. Ce maillon faible, c'est bien souvent l'être humain : mot de passe aisément déchiffrable ou noté à coté de l'ordinateur. Dans la plupart des entreprises, on exige que les mots de passe soient modifiés régulièrement et comportent au moins 8 caractères, mélangeant lettres majuscules, minuscules et chiffres. L'objectif est d'échapper aux logiciels de décodage qui peuvent en peu de temps, balayer tous les mots du dictionnaire. Une protection qui peut s'avérer insuffisante pour l'accès à des applications sensibles.

Le défaut commun à tous les systèmes d'authentification est que l'on identifie un objet (ordinateur, carte, code, ...) et non la personne elle-même. Il est pourtant plus acceptable d'authentifier une personne, plutôt qu'une machine.

Les technologies biométriques de reconnaissance apportent la simplicité et le confort aux utilisateurs, tout en étant superposables avec les systèmes classiques existants. Elles procurent une ergonomie non négligeable dans leur utilisation et sont une brique dans tout système de sécurité actuel et futur. Cette technologie est applicable à un large champ d'applications (contrôle d'accès, gestion horaire, paiement sécurisé sur Internet, login sur ordinateur, etc.). [7]

Il existe deux types d'accès :

- L'accès physique qui désigne tout ce qui est physiquement accessible comme un bâtiment, une salle de laboratoire, un bureau, ...
- L'accès logique qui désigne tout ce qui est virtuellement accessible comme un site internet, un fichier informatique, une application informatique, ...

Les mots de passe :

RSA Security [1] a publiée en septembre 2005 une enquête sur les problèmes rencontrés par l'employé dans la gestion de ses mots de passe ainsi que des risques potentiels pour la sécurité de l'entreprise « J'ai oublié mon mot de passe ».

L'étude réalisée aux Etats-Unis dans 1700 entreprises technologiques montre que plus du quart des personnes interrogées doivent gérer plus de 13 mots de passe. Neuf personnes sur dix s'estiment agacées par la gestion de cette quantité de mots de passe.

Et c'est cette frustration qui peut donner naissance à des comportements dangereux pour la sécurité.

Les entreprises, pour respecter les normes de sécurité, ont été obligées de renforcer l'usage des mots de passe, devenant bientôt un fardeau pour l'utilisateur final. Le mot de passe doit être modifié de plus en plus fréquemment et les caractéristiques auxquelles il doit répondre sont de plus en plus complexes - alliant majuscule, minuscule, chiffre, ponctuation, ...

60% des employés interrogés gèrent plus de 6 mots de passe différents. La plupart d'entre eux (88%) parle de frustration liée à la quantité de codes à retenir.

Les comportements à risque développés par les employés sont nombreux. Certains conservent leurs mots de passe sur un document word dans leur PC. D'autres les notent sur leur PDA ou encore pire sur des feuilles de papier. Le mieux est l'ennemi du bien.

Selon une étude réalisée par une université anglaise, 91% des mots de passe utilisés par des internautes sont connus c'est-à-dire issus de l'environnement familier de la personne et jugés non viables par les spécialistes de cryptage.

- 21% utilisent leur prénom ou celui d'un membre de la famille
- 15% leur date de naissance ou d'anniversaire
- 15% les noms de leurs animaux
- 14% le prénom d'un membre de leur famille
- 7% ont un lien avec une date clé
- 2% utilisent « password »
- 30% des personnes partagent leur mot de passe avec leur partenaire
- 50% seulement affirment être les seuls à connaître leur mot de passe

Qu'est ce que la Biométrie ?

Il existe trois possibilités pour prouver son identité :

1. Ce que l'on possède (carte, badge, document) ;
2. Ce que l'on sait (un nom, un mot de passe) ;
3. Ce que l'on est (empreintes digitales, main, visage, ...) - Il s'agit de la biométrie.

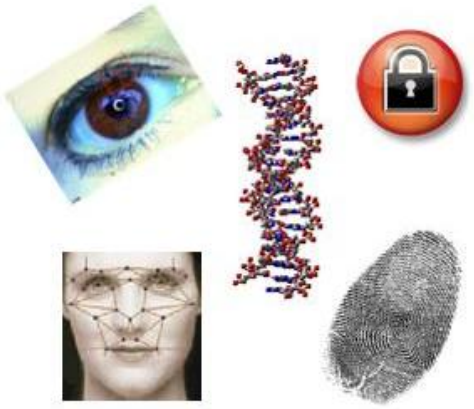
Les 2 premiers moyens d'identification peuvent être utilisés pour usurper l'identité d'un tiers.

1. **RSA Security** est une entreprise inscrite au NASDAQ appartenant à EMC depuis le 14 septembre 2006. Son siège social est à Bedford, au Massachusetts, et la société a des bureaux en Irlande, au Royaume-Uni, à Singapour et au Japon.

Il existe trois catégories de technologies biométriques :

1. **Analyses biologiques** : Odeur, sang, salive, urine, ADN, ...
2. **Analyses comportementales** : La dynamique de la signature (la vitesse de déplacement du stylo, les accélérations, la pression exercée, l'inclinaison), la façon d'utiliser un clavier d'ordinateur (la pression exercée, la vitesse de frappe), la voix, la manière de marcher (démarche), ...
3. **Analyses morphologiques** : empreintes digitales, forme de la main, traits du visage, dessin du réseau veineux de l'œuil, etc.. Ces éléments ont l'avantage d'être stables dans la vie d'un individu et ne subissent pas autant les effets du stress par exemple, que l'on retrouve dans l'identification comportementale.

Les analyses biologiques et morphologiques peuvent être regroupées sous le nom d'analyses (ou biométries) physiologiques.



Les caractéristiques collectées doivent être :
- universelles (exister chez tous les individus),
- uniques (permettre de différencier un individu par rapport à un autre),
- permanentes (autoriser l'évolution dans le temps),
- enregistrables (collecter les caractéristiques d'un individu avec l'accord de celui-ci),
- mesurables (autoriser une comparaison future).
- et si possible infalsifiables, ...

Pourquoi utiliser la biométrie ?

La biométrique est un domaine émergeant où la technologie améliore notre capacité à identifier une personne. La protection des consommateurs contre la fraude ou le vol est un des buts de la biométrie.

L'avantage de l'identification biométrique est que chaque individu a ses propres caractéristiques physiques qui ne peuvent être changées, perdues ou volées. La méthode d'identification biométrique peut aussi être utilisée en complément ou remplacement de mots de passe.

Plusieurs raisons peuvent motiver l'usage de la biométrie :
- **Une haute sécurité** - En l'associant à d'autres technologies comme le cryptage, la carte à puce, ...
- **Confort** - En remplaçant juste le mot de passe, exemple pour l'ouverture d'un système d'exploitation, la biométrie permet de respecter les règles de base de la sécurité (ne pas

inscrire son mot de passe à coté du PC, ne pas désactiver l'écran de veille pour éviter des saisies de mots de passe fréquentes). Et quand ces règles sont respectées, la biométrie évite aux administrateurs de réseaux d'avoir à répondre aux nombreux appels pour perte de mot de passe (que l'on donne parfois au téléphone, donc sans sécurité).
- **Sécurité / Psychologie** - Dans certains cas, particulièrement pour le commerce électronique, l'usager n'a pas confiance. Il est important pour les acteurs de ce marché de convaincre le consommateur de faire des transactions. Un moyen d'authentification connu comme les empreintes digitales pourrait faire changer le comportement des consommateurs.

Les systèmes biométriques suppriment les risques :

	Copie	Vol	Oubli	Perte
Clé	x	x	x	x
Badge	-	x	x	x
Code	x	-	x	-
Biométrie	-	-	-	-

L'usager craint-il la biométrie ?
- Il y a encore quelques années la réponse était « oui ». L'usager potentiel associait la biométrie à police et à fichage étatique.
- Dans un fichier caractérisant un élément biométrique nous concernant, il n'y a pas d'information sur notre vie privée.
- C'est le fichier d'information sur notre personne qui est à mettre en cause, et là, même sans biométrie ce fichier peut exister.
- Aujourd'hui, on se rend bien compte que c'est justement le moyen le plus efficace pour protéger notre bien, qu'il soit matériel ou sous la forme de données informatiques.

Des freins psychologiques existent encore un peu pour l'empreinte digitale, probablement dus à la connotation policière de cette technique. Un utilisateur acceptera assez facilement de se faire reconnaître par son empreinte digitale pour accéder à ses outils personnels comme son PC ou son téléphone, mais il sera réticent à faire le même geste pour accéder à son entreprise.

Pourtant, d'après un sondage réalisé en mai 2005 par IPSOS [2] , les Français approuvent à 75 % la constitution d'un fichier national des empreintes digitales pour lutter contre la fraude et sont 69 % à estimer même que la future carte d'identité biométrique devrait être obligatoire alors que la carte d'identité actuelle n'est que facultative.

L'acceptabilité, par les usagers, d'un système d'identification sera d'autant plus grande que ceux qui doivent l'utiliser sont persuadés qu'il y a quelque chose à protéger, que son utilisation ne présente pas de danger pour la santé et que cela ne permettra pas la collecte d'informations personnelles utilisables à d'autres fins.

La biométrie est-elle concurrente à la carte à puce ?
- Pas du tout, ces deux technologies sont souvent associées.
- Les cartes à puce sont des produits de plus en plus fiables pour sécuriser des informations.
- L'association de la biométrie et de la carte à puce permet d'être certain que l'on est bien le possesseur autorisé de cette carte et des informations qu'elle contient.
- Dans un premier temps, on utilise la mémoire de la carte à puce pour enregistrer son empreinte (pas de base de données).
- Dans le futur, les capteurs d'empreintes et une partie du logiciel de comparaison seront également sur la carte. [7]

2. **IPSOS** est un institut de sondages français et une société internationale de marketing d'opinion, créé en 1975 et dirigée par Didier Truchot et Jean-Marc Lech.

1.2.2 Définition

Les technologies de la biométrie sont définies comme « les méthodes automatisées pour identifier ou authentifier l'identité d'un être vivant en se basant sur une caractéristique physiologique ou comportementale ». [8]

On examinera dans ce qui suit les mots clés de cette définition.

Méthodes automatisées

Le terme dispositif biométrique dans l'industrie de contrôle d'accès implique la présence de trois parties importantes :

1. Un mécanisme pour capturer une image numérique ou analogique d'une caractéristique d'un être vivant.
2. la compression, le traitement et la comparaison de l'image.
3. l'interface avec les systèmes d'applications.

Ces parties peuvent être configurées de plusieurs manières pour différentes situations. Le mot « automatisées »est nécessaire pour notre définition parce que sans lui, on est obligé de décrire une variété de techniques très communes, mais non automatisées et moins fiables, telles une photo ou une empreinte digitale encrée sur un badge de carte d'identité.

La figure 1.12 montre la typologie des techniques d'identification basées sur les caractéristiques humaines, en incluant les technologies biométriques automatisées les plus importantes.

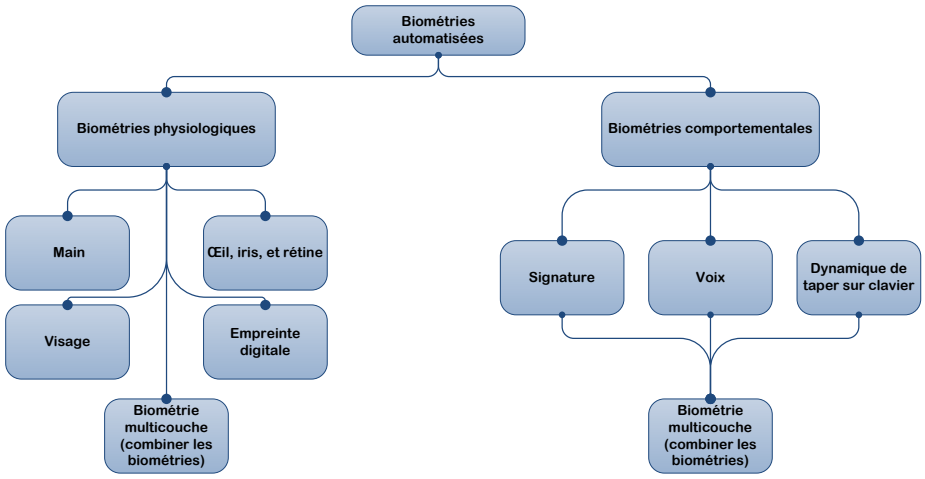


FIGURE 1.12 – Typologie des méthodes biométriques

Identification et authentification

C'est sans doute l'aspect le plus important dans la définition des biométries, car les produits associés à chaque catégorie sont considérablement différents.

Identification : l'identification se présente quand les caractéristiques d'un individu sont choisies à partir d'un groupe d'images conservées. Appelé recherche « one-to-many », la question posée à la machine est « est ce que je vous connais ? ». L'algorithme va chercher dans une base de données et renvoi une liste convenable de candidats après un certain temps. Ces types de produits peuvent coûter entre 40000$ et 1 millions $ selon la configuration.

L'application la plus connue des dispositifs d'identification est l'assistance de l'administration et le renforcement des systèmes de sécurité et de protection des individus et des biens.

Les systèmes AFIS (système d'identification d'empreinte digitale automatisé) peuvent exécuter plus de 100000 essais d'empreinte digitale par seconde.

Authentification : l'authentification se présente quand un individu confirme son identité par la présentation d'un code ou d'une carte. Appelé recherche « one-to-one », la question posée à la machine est « est ce que vous êtes celui que vous prétendez être ? ». Dans ce sens, les caractéristiques de l'individu sont mesurées comparativement à une image conservée dans une base de données. Ces types de produits peuvent coûter entre 100$ et 3000$ et sont utilisés dans les applications incluant les deux contrôles d'accès physique et logique.

Etant donné que la personne qui se présente pour l'authentification donne un code confidentiel ou un mot de passe comme index, le temps de recherche est beaucoup plus rapide que celui de l'identification (100000 concordances par seconde) ; une authentification se produit dans une milliseconde.

Être vivant

Bien que ce terme semble être évident, il est important pour la définition. Une des premières questions qui se posent est « qu'en est-il d'un doigt en latex, d'une bande audio numérique, d'une main de plâtre ou d'un œuil prothétique, etc. ? »La réponse est que beaucoup, mais pas tous les artifices,incluent des méthodes pour déterminer s'il y a bien présence d'une caractéristique vivante. Les méthodes sont quelquefois astucieuses et d'habitude plus simple que prévues.

Caractéristiques physiologiques et comportementales

Le dernier point de la définition est la différence entre les caractéristiques physiologiques et comportementales.

Une caractéristique physiologique est une caractéristique physique relativement stable, comme les traits du visage, l'empreinte digitale, la silhouette de la main, le motif de l'iris ou le motif des vaisseaux sanguins de la rétine. Ce type de mesure est fondamentalement immuable et inaltérable sans contrainte significative.

Une caractéristique comportementale est plus une réflexion du maquillage psychologique d'un individu, bien que les traits physiques généraux, comme la taille et le sexe, aient une influence importante. La signature est la plus commune trace comportementale utilisée dans l'identification. D'autres traces comportementales qui peuvent être utilisées consistent en : comment on tape sur un clavier ou comment on parle. À cause de la variabilité au cours du temps de la plus part des caractéristiques comportementales, beaucoup de machines actualisent leur gabarit biométrique de référence chaque fois qu'il est utilisé. Après beaucoup d'approches réussies, le gabarit peut être de façon significative différent des données originales. La machine sera aussi devenue plus compétente dans l'identification. Généralement l'utilisation régulière des biométries comportementales permet d'avoir de meilleurs résultats.

Les différences entre les méthodes physiologiques et comportementales sont importantes pour plusieurs raisons. D'abord, le degré de variation intra personnelle dans une caractéristique physique est plus petit que celui d'une caractéristique comportementale. Sauf en cas de blessure du doigt, votre empreinte digitale est toujours la même. Une signature, par contre, est tant sous l'influence des actions contrôlables que sous l'influence des facteurs psychologiques moins contrôlables.

Les développeurs des systèmes basés sur la biométrie comportementale, donc, ont une tâche plus dure nécessitant l'adaptation aux variations intra personnelles. Par exemple, il est plus facile de construire une machine qui vous guide à placer votre main toujours dans la même position que de construire des algorithmes qui tiennent compte des états émotionnels. Le problème des machines qui mesurent les caractéristiques physiques est qu'elles ont tendance à être plus grandes, plus chères et peuvent être vues comme menaçantes aux utilisateurs. La biométrie comportementale excelle souvent dans ces domaines. Les deux techniques fournissent de façon significative un niveau supérieur d'identification et de responsabilité que les mots de passe ou les cartes seuls.

A cause de ces différences, on ne peut répondre à tous les besoins par une seule technique. Une compagnie peut même décider d'utiliser différentes techniques dans différentes parties du même système de contrôle d'accès.

1.2.3 Bref panorama des principales biométries et leurs performances

1. **Les performances intrinsèques** :

   Les principaux procédés biométriques ayant émergé sont les suivants :
   - l'ADN
   - la rétine
   - l'iris[3]
   - l'empreinte digitale (et l'empreinte palmaire)
   - la reconnaissance faciale
   - la géométrie du contour de la main
   - la voix
   - l'écriture manuscrite

   Cette liste est établie en fonction de l'ordre de la performance, définie en termes de capacité de discrimination entre individus, allant du plus performant au moins performant.

   Il existe une frontière nette séparant les 5 premières biométries des autres. Pour les cinq premières, la probabilité est très faible, voire quasiment nulle, de trouver sur terre deux individus possédant les mêmes caractéristiques. C'est à dire que la capacité (théorique) de discrimination est au minimum de un sur plusieurs millions d'individus (dans les cas de l'ADN, la rétine et l'iris, elle est plutôt de un sur plusieurs milliards). Pour les autres biométries restantes, la capacité de discrimination n'atteindrait une valeur acceptable que si on limite leur emploi à une population ne dépassant pas quelques milliers d'individus.

   En dehors de toutes considérations tenant à la qualité des produits industriels disponibles sur le marché, le facteur « pourcentage d'individus ne pouvant être traités »affecte les performances intrinsèques d'un procédé biométrique. Ce pourcentage d'exclusions résulte, par exemple, pour la reconnaissance faciale, des habitudes vestimentaires (le port du voile), ou la présence de cheveux et de barbe recouvrant une bonne partie du visage et, pour les empreintes digitales, des personnes handicapées de la main ou pratiquant certains métiers ayant pour conséquence de « raboter »les empreintes des doigts.

   Il semblerait que quelle que soit la biométrie concernée, le pourcentage d'exclusions ne descend jamais en dessous de 1% de la population et pourrait même approcher les 5%.

3. La biométrie de la rétine analyse le fond de l'œuil. L'iris concerne la partie externe avant.

Enfin, certaines biométries laissent des traces qui peuvent être utilisées et traitées à l'insu de la personne concernée : c'est le cas de l'ADN, des empreintes digitales, et peut-être bientôt du visage et de l'iris, si la vidéosurveillance se généralise et si la technologie de ces procédés progresse.

2. **Les performances liées à un produit biométrique spécifique constructeur** :
   Les performances évoquées dans la section précédente concernent les performances intrinsèques, potentielles (théoriques) de chaque procédé biométrique. Toutefois, sur le plan de la réalité du terrain, il est nécessaire de faire intervenir d'autres facteurs ayant une influence notable sur les performances globales d'un procédé biométrique :

   (a) la qualité du capteur
   (b) la qualité du logiciel d'analyse de l'échantillon biométrique ;
   (c) la qualité du logiciel de comparaison ;
   (d) les conditions réelles de mise en œuvre sur le terrain.

   La qualité du capteur : Pour une même biométrie, plusieurs catégories de capteurs sont en général disponibles et dont les performances dépendent largement du prix que l'on est prêt à lui consacrer (on trouve par exemple plusieurs modèles de caméra pour la reconnaissance faciale et, une grande variété entre capteurs capacitif, thermique, optique et peut-être d'autres pour les empreintes digitales).
   Le logiciel d'analyse de l'échantillon biométrique peut donner lieu à des variations non négligeables des performances d'un constructeur à l'autre selon la stratégie adoptée.
   Ainsi, pour prendre l'exemple de l'empreinte digitale, on distingue 2 grandes catégories d'approches, l'une basée sur la recherche des minuties, points caractéristiques remarquables situés sur les traits formant l'empreinte (c'est l'approche traditionnelle) et l'autre, plus récente et promise à un bel avenir, consistant à faire une analyse purement statistique de la répartition des points formant le tracé des empreintes (les minuties n'y jouent aucun rôle particulier) ; ce choix préalable étant fait, chaque constructeur dispose de sa propre méthode soit pour repérer les minuties soit pour faire l'analyse statistique !
   Pour la reconnaissance faciale, il est facile d'imaginer les nombreuses approches possibles pour entreprendre l'analyse des caractéristiques d'un visage, ce qui donnera lieu à des performances disparates en fonction de leurs capacités à prendre en compte des situations diverses comme l'éclairage, l'arrière-plan, le sourire/rictus de la personne, l'angle/l'inclinaison de sa tête, la présence d'une moustache ou d'une barbe, le port des lunettes, le vieillissement etc.
   La qualité du logiciel de comparaison est surtout sensible quand la vérification doit s'effectuer par rapport à une base de données constituée de plusieurs millions d'éléments biométriques de référence.
   Enfin, ***les conditions réelles de mise en œuvre sur le terrain*** sont déterminantes pour l'obtention d'une capture de bonne qualité de l'élément biométrique : disposition des capteurs (un capteur placé trop haut pourrait handicaper les personnes de petite taille), séquence de gestes à effectuer par la personne pour présenter de façon optimum l'élément biométrique au capteur, la plus ou moins bonne volonté des individus contrôlés, etc.

3. **Les performances liées au paramétrage du système** :
   La comparaison entre deux échantillons biométriques d'un même individu, même pris dans un intervalle de temps très court, ne peut jamais donner une égalité parfaite (hormis probablement le cas de l'ADN).

Ainsi, par exemple, pour l'empreinte digitale, le système ne retrouve jamais les mêmes minuties d'une prise d'empreinte à l'autre parce que l'individu ne présente jamais deux fois son doigt de la même façon, en exerçant la même pression sur le capteur, etc. (sans compter le fait que le doigt puisse lui-même avoir entre-temps changé de morphologie ou de consistance).

A cela, il faut ajouter, comme cela à été précédemment développé, les performances liées aux qualités intrinsèques du procédé biométrique (par exemple un pouvoir de discrimination faible), diminuées du pourcentage d' « exclusions », et celles liées aux qualités du système biométrique spécifique fabriqué par le constructeur.

Le cumul de toutes ces incertitudes et causes d'erreur a pour conséquence qu'en toute rigueur, un système de contrôle biométrique ne peut donner, lors d'une comparaison entre deux échantillons biométriques, qu'un résultat sous forme de probabilité de coïncidence.

Puisque le résultat d'une comparaison est toujours une estimation (un score), tous les systèmes biométriques donnent la possibilité de paramétrer le seuil d'acceptabilité :

a) soit en exigeant du système un contrôle strict, en mettant par exemple le seuil à 99,8%, signifiant par-là que 2 échantillons ne seront considérés comme provenant d'un même individu que si le score de similitude est supérieur à 99,8%.

b) soit en étant plus tolérant, en autorisant par exemple que le système réponde positivement si le score de similitude n'est pas en dessous de 95%.

Avec l'option a), la conséquence sera d'augmenter le nombre de « faux rejets », c'est à dire, par exemple lors d'un contrôle, d'augmenter le nombre de refus de personnes qui ne sont pourtant pas en fraude.

Le choix b) aura pour conséquence d'augmenter le taux de « fausses acceptations »c'est à dire d'accepter comme identiques des échantillons biométriques qui en réalité proviennent d'individus différents. La fraude sera plus facile.

Une autre conséquence de cette diminution des performances, plus faibles que ne prévoit la théorie, des dispositifs biométriques est qu'il sera nécessaire, pour un même individu, de procéder à plusieurs captures d'éléments biométriques si l'on désire atteindre un haut niveau d'identification. Il en est ainsi de l'empreinte digitale pour laquelle il est communément admis par les industriels que la marge d'erreur réelle des procédés biométriques, pour une seule empreinte, se situe aux alentours de 1 sur 10000.

1.2.4 Conclusion

On peut constater que la biométrie est une véritable alternative aux mots de passe et autres identifiants. Elle permet de vérifier que l'usager est bien la personne qu'il prétend être. Cette technologie est en pleine croissance et tend à s'associer à d'autres technologies comme la carte à puce.

La fabrication des produits d'authentification est en pleine augmentation, due en l'occurrence à la nécessité croissante du besoin de sécurité de chacun (tant dans le domaine privé que dans le domaine professionnel ou public).

Le coût prohibitif de ces technologies a longtemps freiné leur développement. Aujourd'hui, les organisations (publiques et privées) entrevoient les économies qu'elles réaliseraient à long terme en les utilisant (ex : temps perdu par les services informatiques pour retrouver les mots de passe oubliés).

Pour qu'un système d'authentification soit robuste et paré à toute épreuve l'on peut penser qu'il serait préférable d'associer simultanément plusieurs méthodes d'authentification biométrique en les combinant.

Dans le passé, le traitement automatique (informatisé) de la reconnaissance d'empreintes digitales nécessitait l'utilisation d'importants moyens matériels de traitement. Le coût d'élaboration d'un tel système en cantonnait l'usage à des applications spécifiques et à des organismes très motivés qui y mettaient les moyens (judiciaire, fichier national d'identité, contrôle d'accès haute sécurité).

A présent, les composants possèdent la puissance nécessaire à un traitement de ce type et leur coût ne cesse de décroître.

Les limites de la biométrie :

La biométrie présente malheureusement un inconvénient majeur ; en effet aucune des mesures utilisées ne se révèle être totalement exacte car il s'agit bien là d'une des caractéristiques majeures de tout organisme vivant : on s'adapte à l'environnement, on vieillit, on subit des traumatismes plus ou moins importants, bref on évolue et les mesures changent.

Les fabricants ne recherchent pas uniquement la sécurité absolue, ils veulent quelque chose qui fonctionne dans la pratique. Ils cherchent donc à diminuer le taux de faux rejets (False Rejection Rate, FRR), tout en maintenant un taux relativement bas de fausses acceptations (False Acceptation Rate, FAR). Un système fonctionnel aura un FRR le plus bas possible. D'autre part, une FA est le fait d'accepter une personne non autorisée. Cela peut arriver si la personne a falsifié la donnée biométrique ou si la mesure la confond avec une autre personne. Un système sûr aura un FAR le plus bas possible.

Dans la vie courante, les industriels cherchent principalement à avoir un compromis entre ces 2 taux, FRR et FAR.

De manière générale, les faiblesses de ces systèmes ne se situent pas au niveau de la particularité physique sur laquelle ils reposent, mais bien sur la façon avec laquelle ils la mesurent, et la marge d'erreur qu'ils autorisent.

Chapitre 2

Etude comparative de différentes techniques utilisées pour la reconnaissance de personnes et justification du choix de la reconnaissance de visages

2.1 Principes des systèmes biométriques

2.1.1 Fonctionnement

Comme évoqué ultérieurement, on distingue deux modes d'utilisation distincts d'un système biométrique. On peut vouloir **identifier** une personne parmi un ensemble composé de N individus. Par exemple, on veut repérer un malfaiteur potentiel (à partir d'une liste de criminels identifiés) dans une foule (aéroport, terrain de foot, grande surface, ...). Plus facile est la tâche qui consiste à vérifier l'identité d'une personne, encore appelée **authentification**. Ainsi, face à un individu qui se présente au guichet d'une banque ou à l'entrée d'un bâtiment et qui se prétend être un client répertorié, le système devra simplement prendre une décision d'acceptation ou de rejet de cette personne. [9]

2.1.2 Architecture

En général, on répertorie deux phases dans un système biométrique, une phase d'**apprentissage**, appelée aussi enrôlement, et une phase de **reconnaissance**, ou vérification (figure 2.1). Dans tous les cas, la modalité considérée (par exemple, l'empreinte digitale ou la voix) est enregistrée à l'aide d'un capteur et des données numériques sont alors disponibles (un tableau de pixels, un signal numérique, ...). En général, on ne travaille pas directement sur ces données mais on en extrait d'abord des caractéristiques pertinentes, qui constituent un **gabarit**. Cela présente un double intérêt : le volume d'informations à sauvegarder est plus restreint et l'anonymat dans le stockage de ces données est favorisé. En effet, à partir de ces caractéristiques, il n'est pas possible de revenir au signal original.

Le rôle du module d'apprentissage est de constituer un modèle d'une personne donnée à partir d'un ou de plusieurs enregistrements de la modalité considérée. La plupart des modèles rencontrés sont des modèles statistiques qui permettent de prendre en compte une certaine variabilité dans les données individuelles.

Le module de reconnaissance permet de prendre une décision. Si l'on est en mode identification, le système compare le signal mesuré avec les différents modèles contenus dans la base de données et sélectionne le plus proche. En mode vérification, le système compare le signal mesuré avec un seul des modèles de la base de données et autorise ainsi la personne ou la rejette.

L'identification peut être une tâche très difficile lorsque la base de données contient des milliers d'individus. Les problèmes de temps d'accès deviennent alors cruciaux. [9]

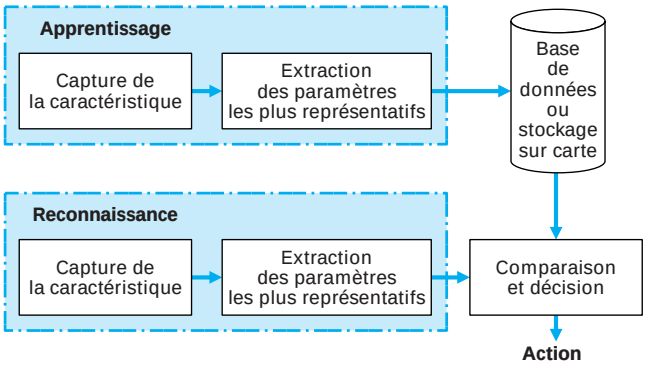


FIGURE 2.1 – Les différents modules d'un système biométrique
Source : [9]

2.1.3 Evaluation

Les évaluations des systèmes biométriques sont souvent réalisées uniquement en terme de performance des systèmes de reconnaissance. Pourtant, il faut garder en mémoire que d'autres facteurs interviennent lors de la mise en œuvre d'un système biométrique. Citons tout d'abord la qualité et la robustesse des capteurs utilisés. Il est assez évident que la qualité des capteurs influe sur les performances des algorithmes de reconnaissance associés. Ce qu'il faudrait donc évaluer, c'est un couple (capteur, algorithme) mais cela reste difficile du fait que ce ne sont souvent pas les mêmes capteurs qui sont utilisés dans les phases d'enrôlement et de test. On est donc plutôt amené à évaluer la résistance d'un algorithme de reconnaissance à l'utilisation de différents types de capteurs (problème d'interopérabilité). Un autre facteur déterminant de l'acceptabilité d'une solution biométrique est la qualité de l'interface de communication associée. Outre le confort d'utilisation, la vitesse d'acquisition et la rapidité du traitement sont des facteurs déterminants, bien souvent non évalués en pratique.

Dans le cas d'un système de vérification, on évalue deux taux d'erreur qui varient en sens contraire : le taux de faux rejet FRR (false rejection rate : taux de rejet d'un utilisateur légitime) et le taux de fausse acceptation FAR (false acceptation rate : taux d'acceptation d'un imposteur). Etant donné un système de vérification, la figure 2.2 représente la distribution théorique des taux de vraisemblance des utilisateurs légitimes et des imposteurs. Les FAR et FRR sont alors représentés, dépendant d'un seuil qui devra être ajusté en fonction des caractéristiques souhaitées pour l'application considérée (haute ou basse sécurité). En effet, plus le seuil est bas, plus le système acceptera d'imposteurs. Plus il est élevé, plus le système sera robuste aux imposteurs mais il rejettera alors plus de vrais utilisateurs. Chaque application nécessite de recalculer ce seuil pour l'adapter à la population spécifique considérée.

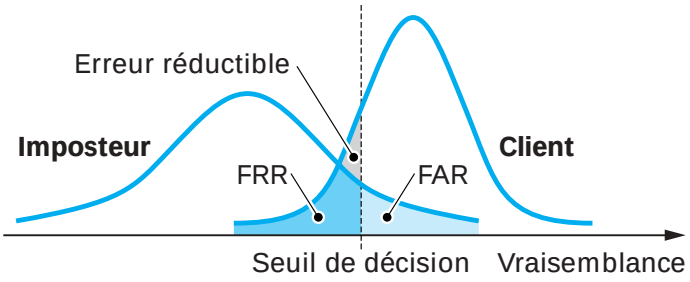


FIGURE 2.2 – Taux de vraisemblance des utilisateurs légitimes et des imposteurs d'un système de vérification biométrique

Source : [9]

La courbe ROC (Receiver Operating Characteristic) de la figure 2.3 permet de représenter la performance d'un système de vérification en termes de couples (FAR, FRR) en fonction des différentes valeurs possibles du seuil. Le taux d'erreur égale EER (equal error rate) correspond au point FAR = FRR et est souvent utilisé pour mesurer la performance du système. [9]

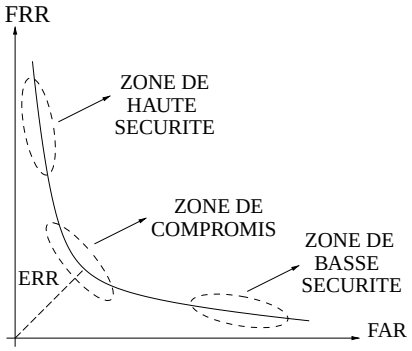


FIGURE 2.3 – Courbe ROC
Source : [10]

2.2 Les différentes modalités

Il existe aujourd'hui une panoplie assez large de modalités biométriques et il en apparaît constamment de nouvelles. En fait, aucune modalité ne permet d'assurer à la fois une précision suffisante et un confort d'utilisation et cela dans toutes les situations d'usage. De plus, quelle que soit la modalité, il existe toujours des personnes réfractaires (mains usées de travailleurs manuels, visages voilés, voix enrouées). Nous ne décrivons ici que les modalités les plus communes, à savoir les empreintes digitales, la géométrie de la main, la parole, l'iris, la rétine, la dynamique de la frappe au clavier, la signature dynamique et la vérification par le visage, laissant de côté d'autres modalités moins classiques (veines de la main, ADN, odeur corporelle, forme de l'oreille, des lèvres, démarche, ...).

2.2.1 Empreintes digitales

L'une des techniques les plus connues du grand public, elle est centenaire.

C'est grâce aux travaux d'Alphonse Bertillon, dans les années 1880, que l'on a commencé à pouvoir identifier des récidivistes sans avoir recours au marquage ou à la mutilation. L'idée d'en faire un instrument d'identification à part entière s'est imposée avec les recherches du Britannique Galton, qui démontra la permanence du dessin de la naissance à la mort, son inaltérabilité et son individualité.

Les minuties des empreintes digitales

La minutie, selon Galton, c'est l'arrangement particulier des lignes papillaires formant des points caractéristiques à l'origine de l'individualité des dessins digitaux (figure 2.4). Arrêts de lignes, bifurcations, lacs, îlots, points, la combinaison des minuties est pratiquement infinie. Dans la pratique judiciaire des pays développés, il faut de 8 à 17 points (mais le plus souvent 12 suffisent) sans discordance pour qu'on estime établie l'identification.

La biométrie par l'empreinte digitale est la technologie la plus employée à travers le monde. Et on voit fleurir des solutions de plus en plus abordables et performantes. D'ici à quelques années, les lecteurs d'empreintes digitales n'étonneront plus personne et seront rentrés dans les mœurs au même titre que le téléphone portable.

FIGURE 2.4 – Eléments caractéristiques visibles sur l'image agrandie d'une empreinte
Source : [9]

2.2.2 Géométrie de la main

Cette technique consiste à déterminer les caractéristiques de la main d'un individu : forme, largeur, longueur, courbure des doigts, etc. C'est une technologie moins contraignante que les empreintes digitales car les saletés et les petites coupures n'entravent pas la reconnaissance.

L'utilisateur doit poser la paume de sa main sur une plaque qui possède des guides afin de l'aider à positionner ses doigts. Une photo de la face de la main est alors prise par un appareil photo numérique ou un scanner. On peut aussi acquérir de l'information sur l'épaisseur de la main à l'aide d'une photo de profil. Etant donné la taille du système d'acquisition, les applications sont limitées. Cependant, la CNIL [1] donne facilement son accord à l'utilisation d'une telle modalité (dans le cas d'accès à des cantines scolaires ou à des lieux réservés par exemple) du fait que cette modalité ne laisse pas de trace (contrairement aux empreintes digitales). Cependant, cette modalité n'est pas stable dans le temps (la forme de la main change avec l'âge) et sa fiabilité est moins importante que celle de l'iris ou de l'empreinte digitale. [9]

La biométrie par la forme de la main est simple à mettre en œuvre, elle est très bien acceptée par les utilisateurs aussi bien pour le contrôle d'accès que le pointage horaire. Elle s'utilise en authentification et a prouvé sa fiabilité dans le temps. Elle s'emploie très bien avec des utilisateurs qui manipulent des produits corrosifs par exemple ; pour ces cas les empreintes digitales risquent fort d'être inutilisables. On compte de nombreuses applications à travers le monde, par exemple sur l'aéroport de San Francisco. [11]

2.2.3 Voix

Cela fait environ 30 ans que les scientifiques se sont penchés sur le problème de la reconnaissance vocale, ou plutôt de la reconnaissance du locuteur puisqu'on utilise plutôt le terme reconnaissance vocale pour les logiciels de dictée vocale (reconnaissance de mots). C'est Texas Instrument qui a été le pionnier dans ce domaine en mettant en place un programme de R&D dès les années 60. Ces recherches ont permis de mettre en place dès les années 70 un système de sécurité basé sur la voix pour les pilotes de l'armée américaine. D'autres sociétés ont aussi participé à l'essor de cette technologie. Citons entre autre : AT&T, IDIAP en France, Sandia National Laboratories, et un grand nombre d'universités en France, Grande Bretagne et Australie.

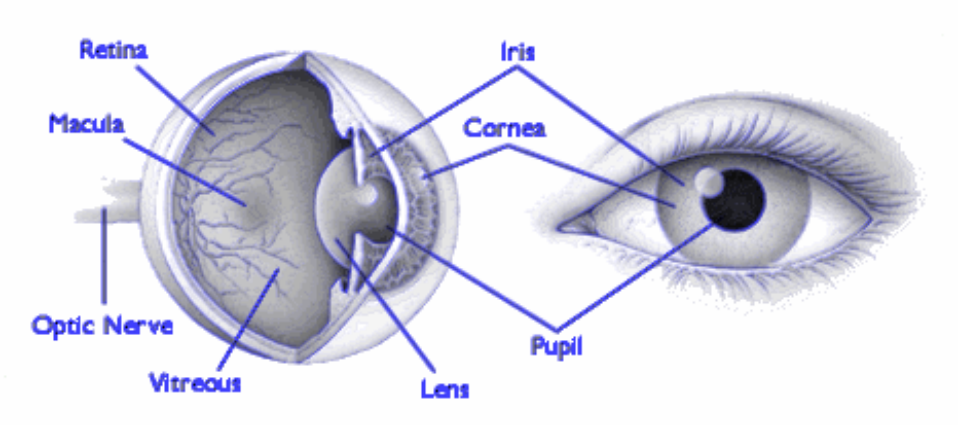
La voix d'une personne se caractérise par beaucoup de paramètres. Chaque personne possède une voix propre que l'on peut analyser par enregistrement avec un micro. Les sons se caractérisent par une fréquence, par une intensité et par une tonalité. Le traitement informatique tient compte des distorsions liées au matériel utilisé, et sait analyser un son de mauvaise qualité tel qu'une transmission téléphonique ou radiophonique. La fatigue, le stress ou un rhume, peuvent provoquer des variations de cette voix. La fraude est possible en enregistrant, à son insu, la voix d'une personne autorisée, à moins d'obliger la personne contrôlée à lire un texte aléatoire.

1. Commission Nationale de l'Informatique et des Libertés (France)

La reconnaissance d'un locuteur offre l'avantage d'être bien acceptée par l'utilisateur, quelle que soit sa culture. De plus, s'il s'agit de sécuriser une transaction téléphonique, la voix est la seule information disponible. On distingue les systèmes à texte prédéterminé (text dependent), où l'utilisateur doit répéter un texte qu'il ne choisit pas, et les systèmes où la personne peut parler librement (text independent). La phase d'apprentissage utilise généralement plusieurs modèles du locuteur pour tenir compte de la variabilité de son discours. La phase de reconnaissance consiste à segmenter le signal de parole en unités qui sont ensuite classées. Ces unités peuvent être des mots ou des phonèmes. La performance est sujette à la qualité du signal, qui dépend de la variabilité de la voix du locuteur dans le temps comme dans le cas de maladie (p. ex. rhume), des états émotionnels (p. ex. angoisse, joie) et de l'âge, des conditions d'acquisition de la voix telles que le bruit et la réverbération, de la qualité des équipements tels que le microphone, sans oublier le fait que différentes personnes peuvent avoir des voix similaires.

La somme des applications ne cesse de s'agrandir chaque jour. Cette technologie est souvent employée dans des environnements où la voix est déjà capturée, comme les centres d'appel et la téléphonie où elle est le moyen biométrique le plus simple et pratique à utiliser.

Pour les 2 techniques suivantes, il faut tout d'abord faire la distinction entre l'iris et la rétine :



Source : American Academy of Ophthalmology

2.2.4 Photographie de l'iris

Les premières traces d'une proposition d'utilisation du motif de l'iris comme moyen de reconnaissance remonte à un manuel d'ophtalmologie écrit par James Doggarts et datant de 1949. On dit même que l'ophtalmologiste Frank Burch en avait émis l'idée dès 1936.

Durant les années 80, l'idée reparut dans différents films de James Bond (Never Say Never Again, 1993), mais elle restait du domaine de la science fiction. Ce n'est donc qu'en 1987 que deux opthtalmologistes (Aran Safir et Leonard Flom) déposèrent un brevet sur cette idée et demandèrent à John Daugman (enseignant à cette époque à l'université de Harvard) d'essayer de trouver un algorithme d'identification basé sur le motif de l'iris. Cet algorithme a été breveté en 1994. Il est la base de tous les systèmes de reconnaissance d'iris actuels.

La personne qui cherche à se faire identifier doit simplement fixer l'objectif d'une caméra qui récupère instantanément le dessin de son iris. L'iris est un motif très dense et qui n'est pas dicté

par les gènes. Chaque œil est unique. Dans toute photographie de l'iris, on compte plus de 200 variables indépendantes, ce qui fait une probabilité très faible de confondre 2 individus. On a obtenu cette méthode grâce à quelques ophtalmologues qui ont remarqué dès les années 80, que la couleur de l'iris peut varier, mais rarement son motif. Cette méthode d'identification évoluera certainement avec le temps, probablement autant que les empreintes digitales, au moins autant que l'évolution des caméras.

Pour capturer l'image de cette membrane colorée, pas besoin d'éclairer la rétine. Par contre, l'éclairage de l'iris pose un problème de reflets, on utilise souvent un éclairage artificiel (diodes DEL) calibré tout en atténuant le plus possible l'éclairage ambiant. L'éclairage est d'autant mieux toléré qu'il peut-être infrarouge, peu visible pour l'œil. (figure 2.5)

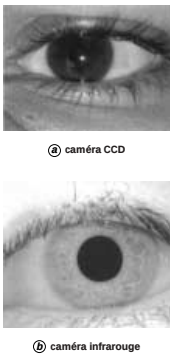


FIGURE 2.5 – Images d'un iris
Source : [9]

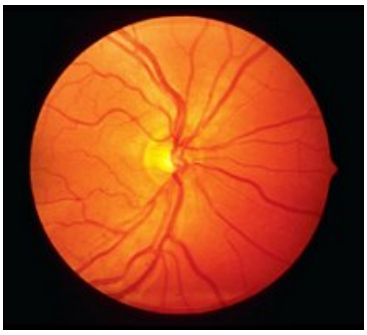
Le système peut être trompé à partir d'une photo ou d'une lentille de contact reproduisant l'iris de la personne dont on souhaite usurper l'identité. Mais la résolution demandée est très importante (distance iris/caméra faible, évolution rapide de la technologie des capteurs CCD/CMOS). De plus il est possible de repérer, par filtrage, que l'iris présenté est constitué d'une suite régulière de points et non d'un motif varié.

Enfin, il existe de nombreuses techniques qui permettent de s'assurer que l'iris présenté est humain (ou très ressemblant) : Si l'on fait varier l'éclairage, le diamètre de la pupille varie. Les temps de latence et vitesse de variation sont mesurables. Il est possible d'éclairer des U.V. à l'I.R. et d'observer les images obtenues. L'œil est opaque dans l'IR lointain (proche du thermique) ainsi qu'aux UV. Ainsi, il semble difficile de fabriquer un faux iris complet (variations de la pupille, réactivité à l'IR, ...).

La biométrie par l'iris est une des technologies (avec la rétine) qui assure un haut niveau de sécurité. L'iris procure une unicité très élevée (1 sur 10 puissance 72) et sa stabilité est étendue jusqu'à la mort des individus, d'où une fiabilité extraordinaire. [11]

2.2.5 La rétine

La rétine est la couche sensorielle de l'œil qui permet la vision. Cette zone est parcourue par des vaisseaux sanguins qui émergent au niveau de la papille optique, où l'on distingue l'artère et la veine centrale de la rétine qui se divisent elles-mêmes en artères et veines de diamètre plus faible pour vasculariser les cellules qui permettent la vision. La grande variété de configurations des vaisseaux sanguins présenterait la même diversité que les empreintes digitales. L'aspect des vaisseaux peut être modifié par l'âge ou la maladie, mais la position respective des vaisseaux reste inchangée durant toute la vie de l'individu. Une caméra est utilisée pour capturer la cartographie des vaisseaux, pour cela il est nécessaire d'illuminer le fond de l'œil.



Source : [12]

- Réputé comme étant le plus fiable moyen biométrique, il souffre d'une réticence psychologique de l'usager. On accepte difficilement l'idée d'un rayon lumineux, même inoffensif, dans l'œil.
- Cette carte vasculaire, propre à chaque individu diffère entre 2 jumeaux et évolue peu avec l'âge.
- Des expériences avec des distributeurs automatiques de billets existent déjà dans certains pays. Contrôle d'accès « haute sécurité », mais le produit restera certainement plus cher que les autres technologies par manque de production de masse.

La biométrie par la rétine procure un haut niveau en matière de reconnaissance. Il est bien adapté pour des applications de haute sécurité (sites militaires, salles de coffres forts, etc.). La disposition des veines de la rétine assure une bonne fiabilité, et une haute barrière contre la fraude. Mais le frein psychologique produit par cette technologie est énorme. Elle est une des raisons de sa faible percée dans les milieux de la sécurité privée. [11]

2.2.6 La dynamique de la frappe (au clavier)

La dynamique de la frappe est propre à chaque individu. Il s'agit en quelque sorte de la graphologie des temps modernes car nous écrivons plus souvent avec un clavier qu'avec un stylo.

Les éléments analysés sont : vitesse de frappe, suite de lettre, temps de frappe, pauses, etc. [11]

2.2.7 Signature dynamique

Chaque personne a un style d'écriture unique. On peut donc définir, à partir de la signature d'une personne, un modèle qui pourra être employé pour effectuer une identification. De plus, la signature est utilisée dans beaucoup de pays comme élément juridique ou administratif. Elle permet de justifier de la bonne foi d'une personne ou de la confondre devant des documents signés.

Le grand avantage des systèmes biométriques à base de signature réside dans la reconnaissance de cette méthode comme une forme acceptable juridiquement pour l'identification des personnes. Cependant, en raison des grandes variations de signature pour une même personne, pour des systèmes tant à base d'analyse statique que dynamique, il est difficile d'atteindre une très haute exactitude d'identification. [11]

2.2.8 Vérification par le visage

Francis Galton instaura les prémices de ce que devrait être la reconnaissance faciale dès 1888 dans son ouvrage « Personnal identification and description »puis y apporta des précisions en 1910 dans un autre ouvrage intitulé « Normalised profiles for classification and recognition ». Quelques publications sont ensuite parues au cours des années 60-70, mais ce n'est qu'a partir du milieu des années 80, lorsque la puissance des ordinateurs est devenue suffisante, que les recherches les plus poussées ont commencé. On a vu alors apparaître des systèmes fonctionnels et des sociétés se sont formées pour exploiter les algorithmes développés dans les laboratoires les plus prestigieux. Depuis le 11 septembre 2002 un intérêt tout particulier est porté sur cette technologie car elle présente de nombreux intérêts, dont celui de pouvoir effectuer une identification à distance. Il ne faut cependant pas oublier les limites technologiques des systèmes actuels qui nécessitent pour l'instant un environnement (éclairage, position de la caméra, etc.) bien contrôlé pour pouvoir pleinement exprimer leurs performances. [11]

Depuis son enfance, tout être humain a appris à reconnaître le visage des personnes qui l'entourent. C'est pourquoi cette modalité nous apparaît comme tout à fait naturelle, d'autant plus que des photos ornent nos documents d'identité. Récemment, différents types de caméras et d'appareils photo, de qualité et de coût variables, sont apparus sur le marché, permettant d'adapter la qualité des images aux conditions d'usage. Pourtant, aujourd'hui, c'est seulement lorsque le sujet est fixe et que les conditions d'environnement sont standards (fond uniforme, éclairage suffisant) que les systèmes informatiques donnent de bons résultats. Si l'acquisition a lieu en environnement naturel, sans contraintes imposées, les performances se dégradent considérablement car les variations personnelles (lunettes, chapeaux, moustaches) ou environnementales (éclairement, éloignement) sont encore mal prises en compte par les systèmes informatiques. Ainsi, les résultats des évaluations récentes proposées par le National Institute of Standards and Technology (NIST) [2] sur les bases de données collectées lors du Facial Recognition Vendor Test (FRVT) montrent que les systèmes sont très sensibles aux variations d'illumination et de pose. C'est certainement le domaine sur lequel la recherche va se concentrer dans les années à venir, compte tenu d'une demande importante du marché.

L'identification fonctionnant sur la reconnaissance faciale, intégrée dans les systèmes de vidéo-surveillance, permettra d'alerter les opérateurs sur la reconnaissance dans l'image d'un individu recherché et préalablement répertorié dans un serveur de base de données. Un suivi de la personne ainsi identifiée, au moyen de caméras situées sur son passage, pourra alors être envisagé. [9]

Plusieurs méthodes de reconnaissance de visages ont été proposées durant ces 30 dernières années, suivant deux grands axes : la reconnaissance à partir d'images fixes et la reconnaissance à partir de séquence d'images (vidéo).

La reconnaissance de visages basée sur la vidéo est préférable à celle basée sur des images fixes, puisque l'utilisation simultanée des informations temporelles et spatiales aide dans la reconnaissance.

La reconnaissance de visages est un gros challenge tellement intéressant, qu'il a attiré les chercheurs des différents domaines : psychologie, identification de modèles, réseaux de neurones, vision d'ordinateur, infographie, etc.

C'est pour cela, que la littérature est si vaste et si diverse. On peut répartir l'ensemble des techniques de reconnaissance de visages, basées sur les images fixes, en trois grandes catégories : les méthodes **locales**, les méthodes **globales** et les méthodes **hybrides**. [13]

2. USA

Le processus de reconnaissance de visages

Tout processus automatique de reconnaissance de visages doit prendre en compte plusieurs facteurs qui contribuent à la complexité de sa tâche, car le visage est une entité dynamique qui change constamment sous l'influence de plusieurs facteurs (figure 2.6).

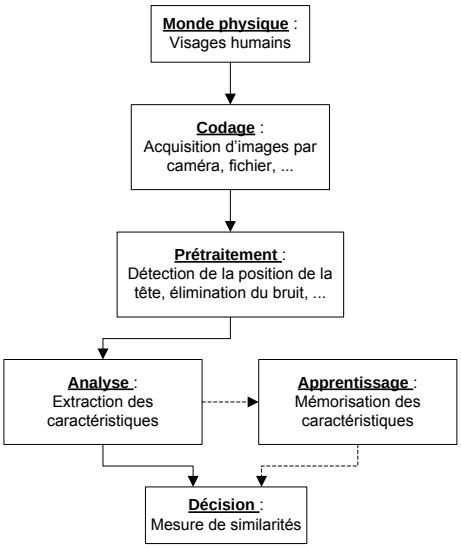


FIGURE 2.6 – Schéma général d'un système de reconnaissance de visages

Dans le **monde physique**, il y a trois paramètres à considérer : L'*éclairage*, la *variation de posture* et l'*échelle*. La variation de l'un de ces trois paramètres peut conduire à une distance entre deux images du même individu, supérieure à celle séparant deux images de deux individus différents.

Le **codage** consiste en l'acquisition de l'image et sa digitalisation, il comporte un risque de bruit et donne lieu à une représentation 2-D (la matrice des niveaux de gris) pour un objet 3-D (le visage).

Dans le **prétraitement** il faut éliminer le bruit par des techniques de traitement et de restauration d'images et procéder à une détection de visages, cette opération est très complexe, surtout si l'image contient plusieurs visages ou si l'arrière plan n'est pas neutre. La restauration d'images ou l'élimination du bruit consiste à compenser les dégradations connues ou estimées et à rétablir la qualité initiale de l'image.

Dans l'étape d'**analyse** (appelée aussi indexation, représentation, modélisation ou extraction de caractéristiques), il faut extraire de l'image les informations qui seront sauvegardées en mémoire pour être utilisées plus tard dans la phase de **décision**. Le choix de ces informations utiles revient à établir un modèle pour le visage, elles doivent être discriminantes et non redondantes.

L'**apprentissage** consiste à mémoriser les représentations calculées dans la phase d'analyse pour les individus connus. Généralement les deux étapes d'**analyse** et d'**apprentissage** sont confondues et regroupées en une seule étape.

La **décision** : Pour estimer la différence entre deux images, il faut introduire une mesure de similarité. [14]

Les classes des techniques de reconnaissance de visages

Méthodes locales : Ce sont des méthodes géométriques, on les appelle aussi les méthodes à traits, à caractéristiques locales, ou analytiques.

L'analyse du visage humain est donnée par la description individuelle de ses parties et de leurs relations. Ce modèle correspond à la manière avec laquelle l'être humain perçoit le visage, c'est à dire, à nos notions de traits de visage et de parties comme les yeux, le nez, la bouche, etc.

La phase d'extraction des points constitue l'étape clé du processus, car la performance du système entier dépend de la précision avec laquelle les informations utiles sont extraites. [14]

L'avantage de ces méthodes réside dans la prise en compte de la particularité du visage en temps que forme naturelle à reconnaître, en exploitant les résultats de la recherche en neuro-psychologie et psychologie cognitive sur le système visuel humain. La difficulté éprouvée lors de la considération de plusieurs vues du visage, ainsi que le manque de précision dans la phase d'«extraction»des points, constituent leur inconvénient majeur et en plus ces techniques sont facilement affectées par l'information non pertinente. [13]

Parmi les méthodes locales, on trouve : *Pure geometry methods*, *Dynamic link architecture*, *HMM*, etc.

Méthodes globales : Les méthodes holistiques appelées aussi méthodes globales, sont des méthodes qui sont normalement utilisées dans la reconnaissance de visage, ces méthodes utilisent la région entière du visage comme entrée à l'algorithme de reconnaissance. Ce sont des techniques très réussies et bien étudiées. Ces méthodes offrent les meilleures performances, mais le problème de stockage des informations extraites lors de la phase d'« apprentissage »reste l'inconvénient majeur.

Parmi les méthodes les plus connues on trouve : *PCA*, *LDA/FLD*, *PDBNN*, etc. [14]

Méthodes hybrides : Elles combinent les deux méthodes citées précédemment, offrant ainsi potentiellement le meilleur des deux types de méthodes.

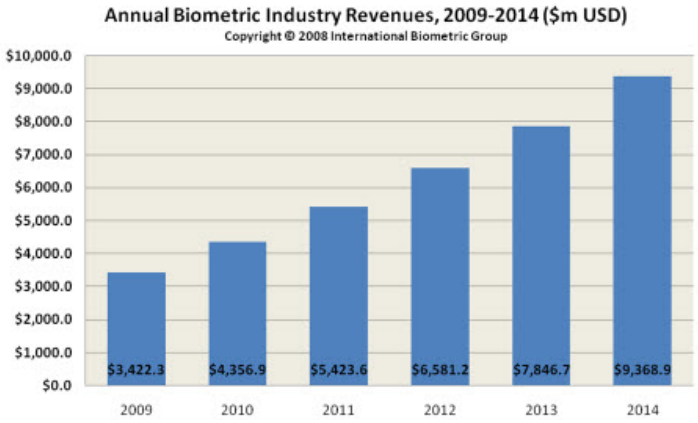
Elles se basent sur le fait que le système de perception humain emploi les caractéristiques locales et la région entière du visage pour l'identification. [15] [16]

Dans cette catégorie, on peut citer les méthodes suivantes : *Hybrid LFA*, *Shape-normalized*, *Component-based*, etc. [13]

2.3 Comparaison des différentes modalités biométriques

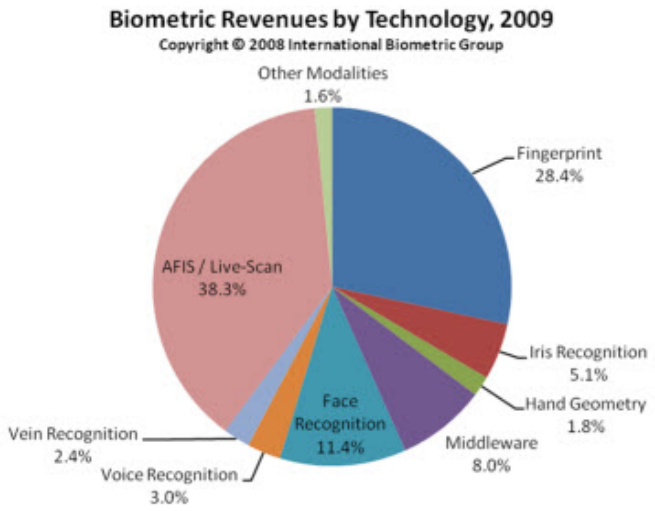
2.3.1 Le marché mondial de la biométrie

IBG (International Biometric Group) [17] édite régulièrement un rapport sur le marché de la biométrie. Cette étude est une analyse complète des chiffres d'affaires, des tendances de croissance, et des développements industriels pour le marché de la biométrie actuel et futur. La lecture de ce rapport est essentielle pour des établissements déployant la technologie biométrique, les investisseurs dans les entreprises biométriques, ou les développeurs de solutions biométriques. [18]



2.3.2 Les parts du marché par technologie

Les empreintes digitales continuent à être la principale technologie biométrique en terme de part de marché, près de 50% du chiffre d'affaires total (hors applications judiciaires). La reconnaissance du visage, avec 12% du marché (hors applications judiciaires), dépasse la reconnaissance de la main, qui avait avant la deuxième place en terme de source de revenus après les empreintes digitales. [18]



2.3.3 Avantages et inconvénients

Dans un premier temps, on peut essayer de calculer les taux d'erreur de différents systèmes dans chaque modalité, mais ces résultats restent limités car seuls certains systèmes dans un nombre restreint d'environnements par application ont été testés. Or, les performances des systèmes dépendent très fortement des conditions de test et, le plus souvent, les systèmes sont évalués en conditions dites de « laboratoire ». La voix, les empreintes digitales et le visage ont fait l'objet des plus nombreux tests et études. Citons en particulier la compétition annuelle NIST de vérification du locuteur, les compétitions FERET et FRVT pour les visages et FVC pour les empreintes tandis que pour l'iris, les évaluations indépendantes sont plus rares. Il faut noter que de nouveaux systèmes apparaissent continuellement et que les performances sont ainsi amenées à s'améliorer. Cette remarque est particulièrement importante par exemple pour la reconnaissance faciale dont les performances sont encore insuffisantes en utilisation réelle.

Plus généralement, la question de l'évaluation des systèmes biométriques fait apparaître d'autres critères que la mesure de la performance tels que la robustesse, l'acceptabilité, la permanence, la facilité de collecte. [9]

Le tableau s'étalant sur les trois pages suivantes récapitulera les forces et faiblesses liés à chacune des technologies utilisées par la biométrie ainsi que leurs domaines d'application. [13]

Biométrie	Avantages	Inconvénients	Domaines d'application
Empreintes digitales	-La technologie la plus éprouvée techniquement et la plus connue du grand public. -Petite taille du lecteur facilitant son intégration dans la majorité des applications (téléphones portables, PC). -Faible coût des lecteurs grâce aux nouveaux capteurs de type "Chip silicium". -Traitement rapide -Bon compromis entre le taux de faux rejet et le taux de fausse acceptation.	-Image "policière" des empreintes digitales. -Besoin de la coopération de l'utilisateur (pose correcte du doigt sur le lecteur). -Certains systèmes peuvent accepter un moulage de doigt ou un doigt coupé (la détection du doigt vivant permet d'éviter ce type d'usurpation).	-En théorie, toutes les applications d'authentification peuvent utiliser les empreintes digitales. Toutefois, le lecteur (capteur) reste exposé à une éventuelle dégradation dans les applications de contrôle d'accès accessible au grand public (distributeur de billets, accès extérieur à des locaux ...). -Contrôle d'accès physique (locaux, machines, équipements spécifiques), contrôle d'accès logique (systèmes d'information).
Géométrie de la main	-Bonne acceptation des usagés -Très simple à utiliser -Le résultat est indépendant de l'humidité et de l'état de propreté des doigts -Fichier "gabarit" de petite taille	-Trop encombrant pour un usage sur le bureau, dans une voiture ou un téléphone -Risque de fausse acceptation pour des jumeaux ou des membres d'une même famille	-Contrôle d'accès à des locaux -Parloirs de prison

Biométrie	Avantages	Inconvénients	Domaines d'application
La voix	-Il est plus facile de protéger le lecteur que dans les autres technologies -Seule information utilisable via le téléphone -Impossible d'imiter la voix -Pas intrusif	-Sensible à l'état physique et émotionnel de l'individu -Fraude possible par enregistrement -Sensible aux bruits ambiants -Taux de faux rejet et fausse acceptation élevés	-C'est le seul moyen pour s'identifier via une liaison téléphonique. -Dans un immeuble d'habitation, on pourra facilement protéger un micro derrière une grille anti-vandalisme.
Photographie de l'iris	-Grande quantité d'information contenue dans l'iris -Vrais jumeaux non confondus	-Aspect psychologiquement invasif de la méthode -L'iris est aisément visible et peut être photographié. Le problème de sécurité est alors lié aux vérifications effectuées lors de la prise de vue. (Problème identique pour les empreintes, la voix, l'oreille,... Mais moins pour la rétine)	-Distributeurs de billets de banque. -Contrôle d'accès physique (locaux, machines, équipements spécifiques), contrôle d'accès logique (systèmes d'information). -En théorie, toutes les applications d'authentification, la caméra est plus exposée qu'un micro (voix) mais moins qu'un capteur tactile (empreintes digitales).

Biométrie	Avantages	Inconvénients	Domaines d'application
La rétine	-L'empreinte rétinienne est peu exposée aux blessures (coupure, brûlure) -Les taux de faux rejet et de fausse acceptation sont faibles -Très difficile, voie impossible, à imiter -La rétine est différente chez les vrais jumeaux -La rétine est stable durant la vie d'un individu	-Système intrusif, il faut placer l'oeil près du capteur -Mauvaise acceptation du public (l'oeil est un organe sensible) -Coût plus important que d'autres technologies -Pas adapté pour un flux de passage important	-Distributeurs de billets de banque. -Contrôle d'accès à des locaux sensibles.
dynamique de la frappe (au clavier)	Non intrusif, geste naturel pour un individu	Dépend de l'état (physique, émotion, fatigue...)	Documents administratif, bancaire, assurance...
Signature dynamique	-La signature écrite sur un document peut être conservée des certains documents -Action qui implique (responsabilité) le demandeur	-Besoin d'une tablette graphique -Sensible aux émotions de l'individu -Pas utilisable pour du conrôle d'accès en extérieur par exemple	Documents administratif, bancaire, assurance...
Vérification par le visage	-Très bien accepté par le public -Ne demande aucune action de l'usager (peu intrusive), pas de contact physique -Technique peu coûteuse	-Technologie sensible à l'environnement (éclairage, position, expression du visage...) -Les vrais jumeaux ne sont pas différenciés -Sensible aux changements (barbe, moustache, lunette, percing, chirurgie...)	-Contrôle d'accès à faible niveau de sécurité -Technologie pouvant être associée avec une autre technologie pour la compléter

TABLE 2.1—Avantages et inconvénients de quelques techniques biométriques

Chapitre 3

Etude théorique de l'algorithme PCA et des réseaux de neurones multicouches

3.1 Algorithme PCA

3.1.1 Introduction

L'algorithme PCA est né des travaux de **M. A. Turk** et **A. P. Pentland** au MIT Media Lab, en 1991. Il est aussi connu sous le nom de *Eigenfaces* car il utilise des vecteurs propres et des valeurs propres (respectivement Eigenvectors et Eigenvalues en anglais). [19]

C'est un algorithme qui identifie des motifs dans des données, et qui exprime les données de façon à accentuer leurs similitudes et différences. Puisqu'il peut être difficile de trouver des motifs dans les données de dimension élevée, où la beauté de la représentation graphique n'est pas disponible, la PCA apparait comme un outil puissant pour l'analyse de ces données.

Un autre avantage majeur de la PCA est qu'une fois que les motifs sont trouvés dans les données, l'image est compressée en réduisant les dimensions, sans perdre beaucoup d'informations.

On illustrera dans ce qui suit par un exemple simple les étapes à suivre pour exécuter l'algorithme PCA sur un ensemble de données. [20]

3.1.2 Etapes d'exécution de l'algorithme PCA

Etape 1 : Avoir des données

Pour notre exemple, nous allons choisir un ensemble de données (figure 3.1) qui a deux dimensions, pour que l'on puisse voir les résultats de l'algorithme sur un graphique.

Data =

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

DataAdjust =

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

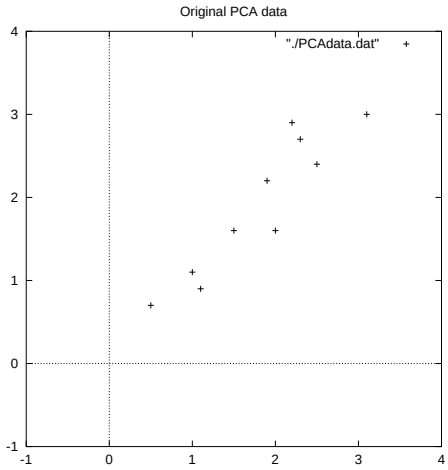


FIGURE 3.1 – L'ensemble de données
Source : [20]

Etape 2 : Enlever la moyenne

Pour que la PCA travaille correctement, on doit soustraire à l'ensemble des valeurs de chacune des dimensions leur moyenne $(\bar{x}, \bar{y})$, ce qui produira un ensemble de données dont la moyenne est nulle (centré).

Etape 3 : Calcul de la matrice de covariance

La variance mesure la dispersion d'une série de valeurs autour de leur moyenne (c'est le carré de l'écart type), c'est une mesure purement unidimensionnelle. Cependant, beaucoup d'ensembles de données ont plus d'une dimension et le but de l'analyse statistique de ces ensembles est habituellement de voir s'il y a un quelconque rapport entre les dimensions.

La covariance permet d'élargir la notion de la variance à deux dimensions. Elle est définie mathématiquement comme suit :

$$\text{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{n-1} \tag{3.1}$$

Où n représente le nombre d'éléments dans les deux dimensions.

Notons que la division se fait sur $n-1$ et non pas sur n. En fait, il a été constaté que si l'ensemble de données est un échantillon d'une population, la division sur $n-1$ donnera une meilleure approximation de la variance de toute la population que celle obtenue par la division sur n. Ce point est explicité par des exemples dans la référence [21].

Si on a un ensemble de données dont la dimension est supérieure à deux, il y aura plus d'une covariance à calculer. De ce fait, pour un ensemble à n dimensions, on peut calculer $\frac{n!}{2\times(n-2)!}$ covariances. Ces covariances sont alors représentées dans une matrice de dimension $(n \times n)$ nommée « matrice de covariance ».

La matrice de covariance à trois dimensions est donnée comme suit :

$$matcov = \begin{pmatrix} \text{cov}(x,x) & \text{cov}(x,y) & \text{cov}(x,z) \\ \text{cov}(y,x) & \text{cov}(y,y) & \text{cov}(y,z) \\ \text{cov}(z,x) & \text{cov}(z,y) & \text{cov}(z,z) \end{pmatrix} \tag{3.2}$$

On remarque que les éléments diagonaux de cette matrice sont les variances des différentes dimensions.

On revient à notre exemple, Comme notre ensemble de données est à deux dimensions la matrice de covariance sera une matrice 2×2. Le résultat est :

$$\text{cov} = \begin{pmatrix} 0.616555556 & 0.615444444 \\ 0.615444444 & 0.716555556 \end{pmatrix} \tag{3.3}$$

Comme les éléments non diagonaux de cette matrice son positifs, on doit s'attendre à ce que les variables x et y augmentent ensemble.

Etape 4 : Calcul des vecteurs propres et valeurs propres de la matrice de covariance

1. **Les vecteurs propres** :

   Comme nous le savons, tant que deux matrices sont de dimensions compatibles, on peut les multiplier.

   Les vecteurs propres sont un cas spécial de cette opération. Considérons, par exemple, les deux multiplications suivantes entre une matrice et un vecteur.

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 11 \\ 5 \end{pmatrix} \tag{3.4}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} \tag{3.5}$$

   Si le résultat de la multiplication d'un vecteur par une matrice, donne un multiple du vecteur original, alors ce dernier sera un vecteur propre.

Dans le premier cas, le vecteur résultant $\begin{pmatrix} 11 \\ 5 \end{pmatrix}$ n'est pas un multiple du vecteur original $\begin{pmatrix} 1 \\ 3 \end{pmatrix}$ donc, ce dernier n'est pas un vecteur propre de la matrice $\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$. Tandis que dans le deuxième cas, le vecteur $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ est un vecteur propre de cette matrice puisque le vecteur résultant est un multiple de ce dernier.

Propriétés :

- Les vecteurs propres ne peuvent être calculés que pour des matrices carrées.
- Une matrice de dimension $(n \times n)$ peut avoir n vecteurs propres au maximum.
- Les vecteurs propres d'une matrice sont orthogonaux. De ce fait, on peut représenter les données d'un ensemble dans une base formée par ces vecteurs propres qu'on normalise, en mettant leurs modules à un.

2. **Les valeurs propres** :

   Les valeurs propres sont étroitement liées aux vecteurs propres.

   Soit l'exemple suivant :

$$2 \times \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \end{pmatrix} \tag{3.6}$$

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \times \begin{pmatrix} 6 \\ 4 \end{pmatrix} = \begin{pmatrix} 24 \\ 16 \end{pmatrix} = 4 \times \begin{pmatrix} 6 \\ 4 \end{pmatrix} \tag{3.7}$$

   Notez que dans l'exemple de la section précédente et celui de cette section, le résultat contient toujours le même facteur de multiplication. Cette grandeur par laquelle le vecteur propre a été multiplié dans le premier exemple du vecteur propre $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$ et dans le deuxième exemple du vecteur $\begin{pmatrix} 6 \\ 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \end{pmatrix} \times 2$ est la même et est égale à 4. C'est ce qu'on appelle la **valeur propre** associée au vecteur propre $\begin{pmatrix} 3 \\ 2 \end{pmatrix}$.

   De ces exemples, on peut voir que les valeurs et vecteurs propres viennent toujours en paire, et que ce qui compte pour un vecteur propre est son orientation et non pas son module.

   On revient à notre exemple, Puisque la matrice de covariance est carrée, nous pouvons calculer ses vecteurs propres et valeurs propres. Ils sont importants, car ils nous indiquent des informations utiles sur nos données. Nous montrerons pourquoi par la suite. En attendant, voici les vecteurs propres et les valeurs propres :

$$Valeurs\ propres = \begin{pmatrix} 0.0490833989 \\ 1.28402771 \end{pmatrix} \tag{3.8}$$

$$Vecteurs\ propres = \begin{pmatrix} -0.735178656 & -0.677873399 \\ 0.677873399 & -0.735178656 \end{pmatrix} \tag{3.9}$$

   Il est important de noter que ces vecteurs propres sont des vecteurs propres unitaires. Leurs modules sont égaux à 1. C'est très important pour la PCA.

Alors, que signifient-ils ? Si on regarde le graphe de données sur la figure 3.2 on verra comment les données sont fortement alignées. Comme prévu de la matrice de covariance, les deux variables augmentent en effet ensemble. Sur le graphe, on a ajouté le tracé des deux vecteurs propres. Ils apparaissent en tant que lignes pointillées diagonales. Comme indiqué dans la section des vecteurs propres, ils sont orthogonaux. Mais, plus important, ils nous fournissent des informations sur les motifs dans les données. Remarquez comment un des vecteurs propres passe par le milieu des points. Ce vecteur propre nous montre comment ces deux dimensions de données sont connexes suivant cette ligne. Le deuxième vecteur propre nous donne l'autre, moins important, motif dans les données, qui indique que tous les points suivent la ligne principale, mais sont loin de part et d'autre de la ligne principale par une certaine quantité.

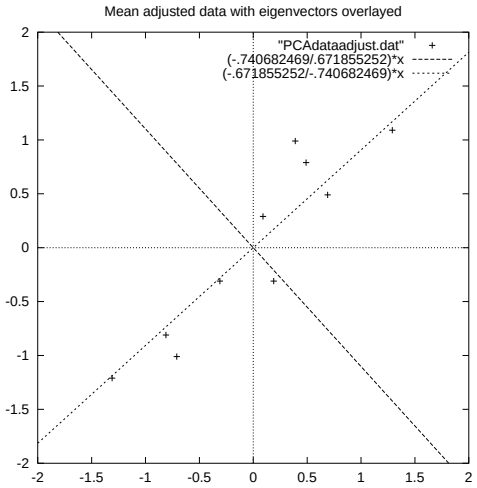


FIGURE 3.2 – Un graphe des données centrées avec les vecteurs propres de la matrice de covariance

Source : [20]

Ainsi, en prenant les vecteurs propres de la matrice de covariance, nous avons pu extraire les lignes qui caractérisent les données. Le reste des étapes se consacre à la transformation des données de sorte qu'elles soient exprimées en termes de ces lignes caractéristiques.

Etape 5 : Choix des composantes et formation du vecteur de caractéristiques (feature vector)

C'est là où la notion de compression de données fait son apparition. En regardant les vecteurs propres et les valeurs propres, on constate que les valeurs propres sont très différentes. En fait, il s'avère que le vecteur propre avec la valeur propre la plus élevée est la **composante principale** de l'ensemble de données.

Dans notre exemple, le vecteur propre avec la valeur propre la plus élevée était celui qui montre la ligne principale de variation des points. C'est la relation la plus significative entre les dimensions de données.

Généralement, une fois que les vecteurs propres de la matrice de covariance sont trouvés, l'étape suivante sera de les mettre en ordre décroissant suivant leurs valeurs propres. Ceci nous donnera les composantes par ordre d'importance. Maintenant, on peut ignorer les composantes les moins significatives. On perd des informations, mais si les valeurs propres sont petites, la perte sera minime. Si on omet quelques composantes, l'ensemble final de données aura moins

de dimensions que l'original.

Ce qui doit être fait maintenant est la formation d'un vecteur de caractéristiques. Ce dernier est construit en prenant les vecteurs propres qu'on veut garder de la liste des vecteurs propres, et en formant une matrice avec ces vecteurs propres dans les colonnes.

$$Vecteur\ de\ caracteristiques = (vp_1 vp_2 \dots vp_p) \tag{3.10}$$

Pour notre ensemble de données, et tenant compte que nous avons deux vecteurs propres, nous avons deux choix. Nous pouvons former un vecteur de caractéristiques avec les deux vecteurs propres :

$$Vecteur\ de\ caracteristiques = \begin{pmatrix} -0.677873399 & -0.735178656 \\ -0.735178656 & 0.677873399 \end{pmatrix} \tag{3.11}$$

Ou bien, on peut choisir d'éliminer le vecteur propre le moins significatif et avoir une seule colonne :

$$Vecteur\ de\ caracteristiques = \begin{pmatrix} -0.677873399 \\ -0.735178656 \end{pmatrix} \tag{3.12}$$

Etape 6 : Formation d'un nouvel ensemble de données

Une fois que nous avons choisi les composantes (vecteurs propres) que nous souhaitons maintenir dans nos données et avons formé un vecteur de caractéristiques, le résultat final sera :

$$Donnees\ finales = Vecteur\ ligne\ de\ caracteristiques \times Vecteur\ ligne\ des\ donnees\ centrees \tag{3.13}$$

Où, *Vecteur ligne de caracteristiques* est le Vecteur de caractéristiques transposé avec le vecteur propre le plus signifiant en haut, et *Vecteur ligne des donnees centrees* est le Vecteur des données centrées transposé. La matrice Données finales est l'ensemble final de données, avec les dimensions suivant les lignes et les données suivant les colonnes.

Qu'est-ce que ceci nous donnera ? Il nous donnera les données originales exclusivement en termes de vecteurs que nous avons choisis. Notre ensemble original de données avait deux axes, x et y, ainsi nos données étaient fonctions d'eux. Il est possible d'exprimer les données en termes de deux axes quelconques qu'on aurait choisis. Si ces axes sont perpendiculaires, alors l'expression est la plus efficace. C'était pour ça qu'il était important que les vecteurs propres soient toujours perpendiculaires entre eux. Nous avons changé nos données et maintenant ils sont fonction de nos deux vecteurs propres. Pour montrer ceci sur nos données, on fait la transformation finale avec chacun des vecteurs de caractéristiques possibles et on représente la transposée du résultat dans chaque cas pour avoir la forme du premier tableau de données. On a également tracé les points finaux pour montrer comment est-ce qu'ils se rapportent aux composantes.

Dans le cas où l'on garde les deux vecteurs propres pour la transformation, nous obtenons les données et le graphe de la figure 3.3. Ce graphe représente fondamentalement les données originales, tournées de sorte que les vecteurs propres soient les axes. C'est compréhensible puisque nous n'avons perdu aucune information dans cette décomposition.

L'autre transformation que l'on peut faire est de prendre seulement le vecteur propre avec la plus grande valeur propre. La table des données résultante est celle trouvée sur la figure 3.4. Comme prévu, elle a seulement une dimension. Si on compare cet ensemble de données à celui résultant de l'emploi des deux vecteurs propres, on notera que cet ensemble de données est exactement la première colonne de l'autre. Ainsi, si on doit tracer ces données, elles seront unidimensionnelles et auront exactement les mêmes positions sur x que les points de la figure 3.3. On a effectivement jeté l'autre axe qui est l'autre vecteur propre.

	x	y
	-.827970186	-.175115307
	1.77758033	.142857227
	-.992197494	.384374989
	-.274210416	.130417207
Transformed Data=	-1.67580142	-.209498461
	-.912949103	.175282444
	.0991094375	-.349824698
	1.14457216	.0464172582
	.438046137	.0177646297
	1.22382056	-.162675287

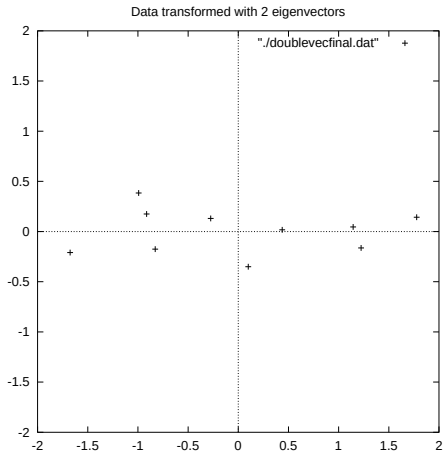


FIGURE 3.3 – La table de données après application de l'algorithme PCA en utilisant les deux vecteurs propres et un graphe pour les nouveaux points des données

Source : [20]

Transformed Data (Single eigenvector)

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056

FIGURE 3.4 – Les données après transformation en utilisant seulement le vecteur propre le plus significatif

Source : [20]

Ainsi qu'avons-nous fait ici ? Fondamentalement nous avons transformé nos données de sorte qu'elles soient exprimées en termes de motifs existants entre eux, où les motifs sont les lignes qui décriront le plus étroitement les rapports entre les données. C'est utile parce que nous avons maintenant classifié nos points de données comme une combinaison des contributions de chacune de ces lignes. Initialement, nous avons les axes x et y. Malheureusement, les valeurs x et y de chaque point de donnée ne nous indiquent pas exactement comment ce point est relié aux autres points de l'ensemble des données. Maintenant, les valeurs des points de données nous indiquent exactement (c.-à-d. au-dessus/au-dessous) les lignes de tendance des points de l'ensemble des données. [20]

3.1.3 Application à la reconnaissance de visages

Cette section décrira la manière avec laquelle la PCA est employée dans la reconnaissance de visages.

En théorie de l'information, nous voulons extraire l'information caractéristique d'une image de visage, pour l'encoder aussi efficacement que possible afin de la comparer à une base de données de modèles encodés de manière similaire ; l'algorithme PCA est un bon outil qui permet d'atteindre ce but.

Une image $I_{i_{(m,n)}}$ est traitée comme un vecteur $\Gamma_{i_{(m\times n,1)}}$dans un espace vectoriel de grande dimension ($N = m \times n$), par concaténation des colonnes :

$$I_1 = \begin{pmatrix} a_{1,1} & \cdots & a_{1,m} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,m} \end{pmatrix} \Rightarrow \Gamma_1 = \begin{pmatrix} a_{1,1} \\ \vdots \\ a_{n,1} \\ \vdots \\ a_{1,m} \\ \vdots \\ a_{n,m} \end{pmatrix} \tag{3.14}$$

La formule 3.14 représente le passage d'une image vers un vecteur dans un espace vectoriel de grande dimension.

Les coefficients $a_{(i,j)}$ représentent les valeurs des pixels en niveau de gris, codés de 0 à 255.

Après avoir rassemblé les M images dans une unique matrice, nous obtenons une *matrice d'images* Γ, où chaque colonne représente une image Γ_i :

$$\Gamma = \begin{pmatrix} a_{1,1} & b_{1,1} & \cdots & z_{1,1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n,1} & b_{n,1} & \cdots & z_{n,1} \\ \vdots & \vdots & \cdots & \vdots \\ a_{1,m} & b_{1,m} & \cdots & z_{1,m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n,m} & b_{n,m} & \cdots & z_{n,m} \end{pmatrix} \tag{3.15}$$

Ce qui nous donnera un point de départ pour l'algorithme PCA.

On calcule l'image moyenne Ψ de toutes les images collectées. Cette image peut être vue comme le *centre de gravité* du jeu d'images :

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i \tag{3.16}$$

On ajuste ensuite les données par rapport à la moyenne. L'image moyenne est alors soustraite de chaque image avec la formule suivante :

$$\Phi_i = \Gamma_i - \Psi, \quad \mathrm{i} = 1 \ldots M \tag{3.17}$$

On calcule ensuite la matrice de covariance du jeu de données. Cette matrice peut être vue comme une *matrice de moments d'ordre 2* :

$$C = \sum_{i=1}^{M} \Phi_i \Phi_i^T = AA^T, \quad A = [\Phi_1 \Phi_2 \ldots \Phi_M] \tag{3.18}$$

La prochaine étape consiste à calculer les vecteurs propres et les valeurs de cette matrice de covariance C de taille $(N \times N)$, c'est-à-dire de l'*ordre de la résolution d'une image.*

Le problème est que cela peut parfois être très difficile et très long ! En effet, si $N > M$ (si la résolution est supérieure au nombre d'images), il y aura seulement $M - 1$ vecteurs propres qui contiendront de l'information (les vecteurs propres restants auront des valeurs propres associées nulles). Par exemple, pour 100 images de résolution 320×240, nous pourrions résoudre une matrice L de 100×100 au lieu d'une matrice de 76800×76800 pour ensuite prendre les combinaisons linéaires appropriées des images Φ_i. Le gain en temps de calcul serait considérable ! Typiquement, nous passerions d'une complexité de l'ordre du nombre de pixels dans une image à une complexité de l'ordre du nombre d'images. **Voici comment procéder pour accélérer les calculs** :

Considérons les vecteurs propres e_i de $C = AA^T$, associés aux valeurs propres λ_i :

$$Ce_i = \lambda_i e_i \tag{3.19}$$

Les vecteurs propres v_i de $L = A^T A$, associés aux valeurs propres λ_i sont tels que :

$$L\nu_i = \mu_i \nu_i \tag{3.20}$$

Soit :

$$A^T A\nu_i = \mu_i \nu_i \tag{3.21}$$

En multipliant à gauche par A des deux côtés de l'égalité, nous obtenons :

$$AA^T A\nu_i = A\mu_i \nu_i \tag{3.22}$$

Puisque $C = AA^T$, nous pouvons simplifier :

$$C\,(A\nu_i) = \mu_i (A\nu_i) \tag{3.23}$$

De 3.19 et 3.23, nous voyons que Av_i et μ_i sont respectivement les vecteurs propres et les valeurs propres de C :

$$\begin{cases} e_i = A\nu_i \\ \lambda_i = \mu_i \end{cases} \tag{3.24}$$

Nous pouvons donc trouver les valeurs propres de cette énorme matrice C en trouvant les valeurs propres d'une matrice L beaucoup plus petite. Pour trouver les vecteurs propres de C, il suffit juste de pré-multiplier les vecteurs propres de L par la matrice A.

Les vecteurs propres trouvés sont ensuite ordonnés selon leurs valeurs propres correspondantes, de manière décroissante. Plus une valeur propre est grande, plus la variance capturée par le vecteur propre est importante. Cela implique que la majeure partie des informations est contenue dans les premiers vecteurs propres.

On sélectionne après les k meilleurs vecteurs propres (ceux avec les k plus grandes valeurs propres). On définit alors un espace vectoriel engendré par ces k vecteurs propres, que l'on appelle l'**espace des visages** E_v « Face Space ».

Les images originales peuvent être reconstituées par combinaison linéaire de ces vecteurs propres. Les représentations graphiques de ces vecteurs rappellent un peu des images fantômes, chacune mettant en avant une partie du visage, on les appelle **eigenfaces** (figure 3.5).

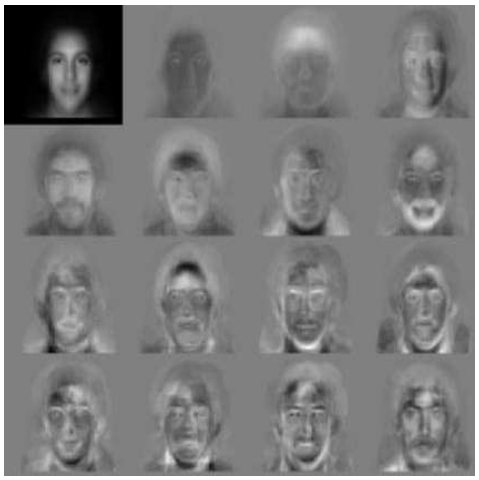


FIGURE 3.5 – Image moyenne et les 15^{res} eigenfaces
Source : [19]

Nous allons maintenant projeter nos images de départ sur E_v. Une image Γ_i est alors transformée en ses composantes eigenfaces par une simple opération de projection vectorielle :

$$w_k = e_k^T(\Gamma_i - \Psi), \quad k = 1, \dots M' \tag{3.25}$$

Les vecteurs w_k sont appelés poids et forment une matrice $\Omega^T = [w_1, w_2, \dots w_{M'}]$ qui décrit la contribution de chaque eigenface dans la représentation de l'image d'entrée.

La matrice Ω^T alors utilisée pour trouver quelle est, parmi un nombre prédéfini de classes, celle qui décrit le mieux une image d'entrée.

La méthode la plus simple pour déterminer quelle classe de visages fournit la meilleure description d'une image d'entrée est de trouver la classe de visages k qui minimise la distance euclidienne.

$$\epsilon_k^2 = \| \Omega - \Omega_k \|^2 \tag{3.26}$$

Où Ω_k un vecteur qui décrit la K^e classe de visage.

Un visage appartient à une classe k quand le minimum ϵ_k est en dessous d'un certain seuil θ_k. Dans le cas contraire, le visage est classé comme étant *inconnu* et peut éventuellement être utilisé pour créer une nouvelle classe de visage. La création de la matrice de poids Ω^T équivalente à la projection du visage original sur E_v. Puisque la distance ϵ entre l'image de visage et E_v est simplement le carré de la distance entre l'image d'entrée réajustée par rapport à la moyenne $\Phi = \Gamma - \Psi$ et $\Phi_f = \sum_{i=1}^{M'} w_i e_i$ sa projection sur E_v est $\epsilon^2 = \| \Phi - \Phi_f \|^2$.

Il y a alors quatre possibilités pour une image d'entrée d'être reconnue ou non (figure 3.6).

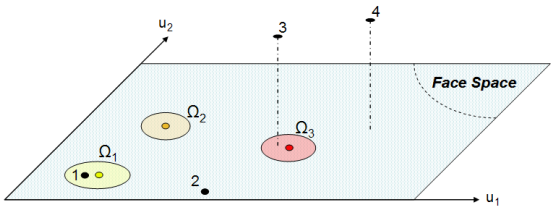


FIGURE 3.6 – Une version simplifiée de E_v illustrant les quatre résultats de la projection d'une image sur E_v. Dans ce cas, il y a deux vecteurs propres ($u1$ et $u2$) et trois classes d'individus connus ($\Omega 1, \Omega 2, \Omega 3$)

Dans le cas 1, un individu est reconnu et identifié. Dans le cas 2, un individu inconnu est présent. Les deux derniers cas (3 et 4) indiquent que l'image n'est pas une image de visage. Pour le cas 3, l'image est éloignée de E_v mais la projection est proche d'une classe connue, on parle alors de fausse acceptation. Ces remarques sont résumées dans le tableau 3.1

	Espace des visages (E_v)	Classe de visage
Cas1	proche	proche
Cas2	proche	éloigné
Cas3	éloigné	proche
Cas4	éloigné	éloigné

TABLE 3.1 – Les quatre possibilités qui apparaissent lors de la phase de reconnaissance

3.1.4 Conclusion

L'algorithme PCA est une méthode globale utilisant en premier lieu les niveaux de gris des pixels d'une image. Sa simplicité à mettre en œuvre contraste avec une forte sensibilité aux changements d'éclairement, de pose et d'expression faciale. Néanmoins, le PCA ne nécessite aucune connaissance à priori sur l'image.

Le principe selon lequel on peut construire un sous-espace vectoriel en ne retenant que les « meilleurs »vecteurs propres, tout en conservant beaucoup d'information utile, fait du PCA un algorithme efficace et couramment utilisé en réduction de dimensionnalité.

Enfin, l'étude théorique de l'algorithme PCA est très pédagogique et permet d'acquérir de solides bases pour la reconnaissance 2D du visage. C'est un algorithme incontournable ! [19]

3.2 Réseaux de neurones multicouches

3.2.1 Introduction

Un réseau de neurones artificiel est une structure composée d'entités capables de calcul et interagissant entre eux, les neurones. Il permet de traiter, par le biais de l'informatique, des problèmes de différentes natures que les outils classiques ont du mal à résoudre. En effet, son fonctionnement s'inspire de celui des cellules neuronales animales, et est donc différent des méthodes de calcul analytiques que l'on utilise ordinairement. Il s'avère très puissant dans des problèmes de reconnaissance, classification, approximation ou prévision.

Les sections suivantes ont été tirées de la référence [22].

3.2.2 Le modèle du neurone

Neurone biologique

En biologie, un neurone est une cellule nerveuse dont la fonction est de transmettre un signal électrique dans certaines conditions. Il agit comme un relai entre une couche de neurones et celle qui la suit. Les caractéristiques des neurones sont encore mal connues (et font l'objet de recherches) mais on connait leur principe d'action.

Le corps d'un neurone est relié d'une part à un ensemble de dendrites (entrées des neurones) et d'autre part à un axone, partie étirée de la cellule, qui représentera pour nous sa sortie. Le neurone étudié est connecté aux neurones qui l'environnent : il reçoit au niveau de ses dendrites les signaux électriques des neurones « en amont », propagés par les axones de ces derniers. Les charges électriques s'accumulent dans le neurone jusqu'à dépasser un certain seuil : à ce moment la transmission du signal électrique se déclenche via son axone vers d'autres neurones « en aval ».

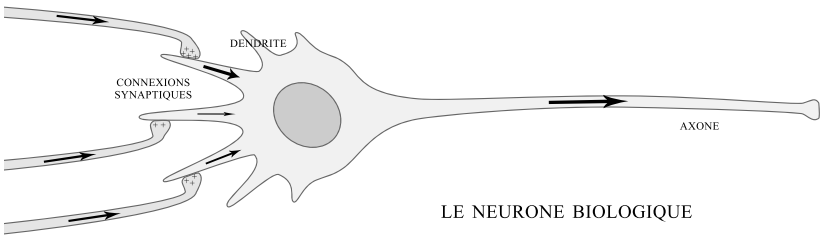


FIGURE 3.7 – Le neurone biologique

On remarque que les liaisons axone/dendrite entre deux neurones (connexions synaptiques) ne sont pas toutes de la même efficacité. Ainsi l'entrée associée à une certaine dendrite du neurone pourra avoir plus d'influence qu'une autre sur la valeur de sortie. On peut représenter la qualité de la liaison par un poids, sorte de coefficient s'appliquant au signal d'entrée. Le poids sera d'autant plus grand que la liaison est bonne. Un poids négatif aura tendance à inhiber une entrée, tandis qu'un poids positif viendra l'accentuer.

Neurone formel

Pour reproduire le neurone biologique, on se sert d'un modèle mathématique : le neurone formel. Il doit être capable de :

- Recevoir en entrée différentes informations provenant des neurones environnants,
- Analyser ces informations, de manière à envoyer en sortie une réponse,
- Ajuster cette réponse avant de l'envoyer aux neurones suivants.

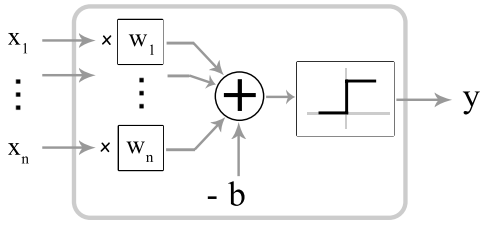


FIGURE 3.8 – Structure générale d'un neurone formel

Il est donc tout naturel d'assimiler un neurone à un triplet :
$(\overrightarrow{poids},\ biais,\ fonction\ d'activation\ f)$

On multiplie chaque valeur d'entrée par la composante des $\overrightarrow{poids}$ correspondante, ce qui revient à faire le produit scalaire $\overrightarrow{entrées} \cdot \overrightarrow{poids}$,

- On compare la valeur obtenue à une valeur de référence : le biais, ce qui revient à soustraire le scalaire biais,
- Enfin on applique la fonction d'activation à cette différence ; la fonction d'activation est souvent utilisée de façon à avoir une sortie comprise entre 0 et 1. Par exemple dans le cas d'une fonction d'activation de type seuil, la sortie sera :

$$\begin{cases} 0 & si\ \overrightarrow{entrées} \cdot \overrightarrow{poids} - biais \leq 0 \\ 1 & si\ \overrightarrow{entrées} \cdot \overrightarrow{poids} - biais > 0 \end{cases} \tag{3.27}$$

On peut donner quelques exemples de fonctions d'activation utilisées pour les réseaux de neurones. Elles ressemblent le plus souvent à des fonctions tout ou rien :

Nom	Valeur	Représentation
Seuil	$f(x) = 0 \quad \text{si} \quad x \leq 0$ $f(x) = 1 \quad \text{si} \quad x > 0$	
Seuil symétrique	$f(x) = -1 \quad \text{si} \quad x < 0$ $f(0) = 0$ $f(x) = 1 \quad \text{si} \quad x > 0$	
Linéaire saturée	$f(x) = 0 \quad \text{si} \quad x < 0$ $f(x) = x \quad \text{si} \quad 0 \leq x \leq 1$ $f(x) = 1 \quad \text{si} \quad x > 1$	
Linéaire	$f(x) = x$	
Sigmoïde	$f(x) = \frac{1}{1+e^{-x}}$	
Tangente hyperbolique	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	

FIGURE 3.9 – Quelques exemples des fonctions d'activation

3.2.3 Structure d'un réseau neuronal

En superposant dans une même couche, la couche 1, s_1 neurones formels à e_1 entrées, on obtient un système admettant e_1 entrées et s_1 sorties (chaque neurone produisant une sortie scalaire). Puis lorsqu'on concatène N couches connectées les unes à la suite des autres ($s_{n-1} = e_n$: le nombre d'entrées d'une couche étant égal au nombre de sorties de la précédente), on obtient un réseau de N couches de neurones. La N^e couche est appelée couche de sortie, tandis que les couches précédentes sont des couches cachées.

Le choix du nombre de couches et de neurones dépend du problème que l'on souhaite résoudre. En effet un réseau sera apte à traiter ce problème si les données de ce dernier fournissent e_1 variables d'entrées et nécessitent s_N variables en sortie.

Un exemple : on souhaite classer des images de $5 \times 5 = 25$ pixels dans 2 catégories, selon des critères dépendant de la couleur et de la disposition de ces pixels. La première couche de notre réseau devra comporter 25 entrées, une pour la valeur de chaque pixel, et la couche de sortie fournira 2 valeurs : les « affinités »respectives qu'aura l'image avec les deux catégories (si la première affinité est la plus grande, l'image sera classée dans la catégorie 1).

3.2.4 Entraînement d'un réseau

La particularité du réseau de neurones comme outil mathématique repose sur sa capacité d'apprentissage : grâce à un processus d'entraînement, on peut améliorer les performances du réseau, c'est-à-dire qu'en réponse à une certaine entrée, il fournisse la « bonne »sortie, la sortie qu'on voudrait obtenir.

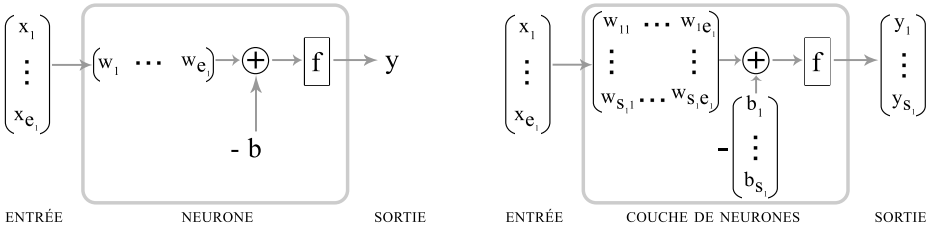
Le principe est le suivant : on peut modifier les caractéristiques du réseau (les poids de ses liaisons par exemple) en réaction aux stimuli extérieurs qu'on lui soumet (les valeurs d'entrée), de manière à ce qu'il réagisse différemment si un même stimulus lui est appliqué ultérieurement. Le réseau s'améliore ainsi car à chaque erreur qu'il fait, il subit une correction qui le fait réagir différemment s'il est confronté à la même situation. Le but étant qu'une fois l'apprentissage terminé, le réseau effectue la tâche pour laquelle il a été conçu sans se tromper, c'est-à-dire qu'il fournisse pour chaque stimulus d'entrée la « bonne »sortie, la sortie désirée par l'opérateur. L'apprentissage n'est bien sûr pas réalisé par l'opérateur par modification des paramètres du réseau à la main, mais par un algorithme d'entraînement. De nombreux algorithmes existent à ce jour, que l'on peut séparer en deux types :

- **Les apprentissages supervisés** : un « professeur » fournit un grand nombre de couples $(\overrightarrow{entrée}, \overrightarrow{sortie}_{désirée\ pour\ cette\ entrée})$, et la correction s'effectue selon l'erreur obtenue pour chaque couple $(\overrightarrow{erreur} = \overrightarrow{sortie}_{obtenue} - \overrightarrow{sortie}_{désirée})$. Si l'apprentissage est efficace, la norme de l'erreur diminue globalement. On l'appelle aussi apprentissage par des exemples.
- **Les apprentissages non-supervisés** : sans professeur définissant la sortie désirée pour une entrée donnée, donc sans signal d'erreur, le réseau apprend par lui-même à classer des entrées similaires en trouvant des points communs aux stimuli appliqués. La répétition de la correction en réaction à des stimuli en entrée constitue le processus d'apprentissage. Il faut parfois des milliers d'itérations de l'algorithme pour arriver à un résultat.

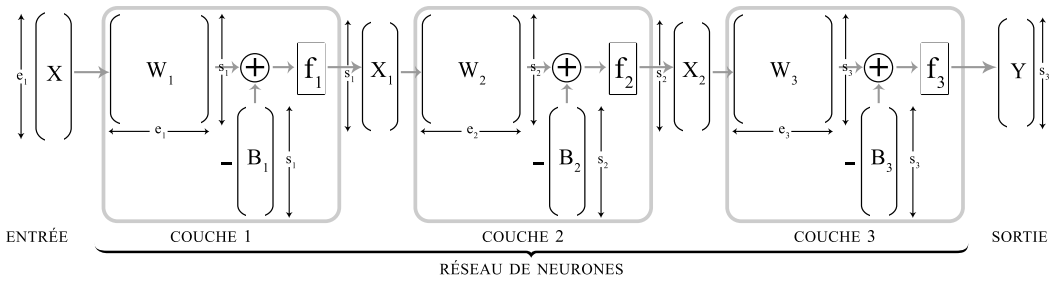
3.2.5 Modélisation mathématique du réseau

L'effet d'un neurone à e_1 entrées peut être modélisé par une forme linéaire de $\mathbb{R}^{e_1}$ dans $\mathbb{R}$ (qui multiplie les composantes de l'entrée par des poids w bien choisis), à laquelle on soustrait un scalaire biais et l'on applique enfin une fonction d'activation f (de type seuil, tangente hyperbolique, etc. a priori non linéaire) au résultat.

L'action d'une couche de s_1 neurones est donc assimilable à une application linéaire de $\mathbb{R}^{e_1}$ dans $\mathbb{R}^{s_1}$, que l'on représentera ici par une matrice à e_1 colonnes et s_1 lignes : une entrée à e_1 composantes aura pour image un vecteur à s_1 composantes (une pour chaque neurone). On soustrait à cette première sortie un biais $\overrightarrow{B_1}$ qui est cette fois-ci vectoriel, et l'on applique une fonction d'activation f_1 à chaque composante du vecteur obtenu.



Un réseau de neurones tout entier (N couches empilées les unes à la suite des autres) est donc une succession de N matrices de tailles $e_i \times s_i$ $(1 \leq i \leq N)$, auxquelles on associe, à chaque fois, un vecteur biais $\overrightarrow{b_i}$ et une fonction d'activation f_i. Par souci de simplicité, on prendra la même fonction d'activation f_i pour tous les neurones d'une même couche. En revanche, utiliser des fonctions différentes pour les différentes couches peut s'avérer assez intéressant (selon l'usage du réseau). Exemple de l'action de la couche 1 :



$$\begin{array}{ccccc} X & \mapsto & W_1 \cdot \mathrm{X} - \mathrm{B1} & \mapsto & f_1\left(W_1 \cdot \mathrm{X} - \mathrm{B}_1\right) = \mathrm{X}_1 \\ \mathbb{R}^{e_1} & \mapsto & \mathbb{R}^{s_1} & \mapsto & \mathbb{R}^{s_1} \end{array} \tag{3.28}$$

Remarque : Chaque couche a ses propres paramètres (nombre d'entrées, nombre de sorties, fonction d'activation) ; mais puisque chaque couche prend en entrée les valeurs de sortie de la couche précédente, on doit toujours avoir $e_i = s_{i-1}$.

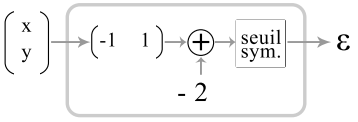
3.2.6 Le perceptron multicouche

Le perceptron multicouche est un type de réseau de neurones parmi les plus utilisés. Son fonctionnement est le suivant : l'information se propage de couche en couche, suivant le modèle décrit plus haut, toujours dans le même sens jusqu'à la sortie. On donne au réseau l'information sous forme d'un vecteur de $\mathbb{R}^{e_1}$ et on récupère l'information traitée sous forme d'un vecteur de $\mathbb{R}^{s_N}$.

Dans certains réseaux autres que le perceptron, plus complexes, l'information ne circule pas en sens unique : il peut y avoir rétroaction.

Un exemple très simple : on considère un perceptron monocouche, dont la fonction sera de détecter si un point du plan $\mathbb{R}^2$ est au-dessus ou en dessous de la droite d'équation $y = x + 2$.

On veut :
- **En entrée**, les coordonnées x et y du point de $\mathbb{R}^2 (\Rightarrow e_1 = 2)$
- **En sortie**, la valeur $+1$ si le point est au-dessus de la droite, -1 s'il est en dessous $(\Rightarrow s_1 = 1)$.



Utilisons alors la fonction d'activation *seuil symétrique*, qui à tout réel positif associe $+1$, à tout réel négatif -1 et à 0 lui-même ; un biais $b = 2$; et une matrice de poids $W = (-1 \quad 1)$.

Si notre point est au dessus de la droite, c'est-à-dire si $y > x + 2$, alors $W \cdot X - b = -x + y - 2 > 0$ et la fonction d'activation donnera $\epsilon = +1$. Si le point sur la droite, $y = x + 2$ et on obtient $\epsilon = 0$. Si le point est en dessous, $y < x + 2$ et alors $\epsilon = -1$.

Cet exemple de réseau à un neurone est peu intéressant : on a du déterminer nous-mêmes W et b tels qu'il fonctionne. Le principal intérêt d'un réseau de neurones est sa capacité d'auto-apprentissage.

3.2.7 Apprentissage par rétropropagation du gradient

Soit un réseau de type perceptron, dont les N couches ont chacune un nombre d'entrées, de sorties, et une fonction d'activation bien déterminés, mais dont les poids et biais sont initialement inadaptés (les matrices W_i et les vecteurs B_i sont remplis de coefficients aléatoires par exemple). Le réseau est ignorant et dépourvu d'expérience : pour une entrée donnée, il ne fournit pas la sortie qu'on souhaiterait obtenir.

Pour rendre le réseau performant, et donc utilisable sans qu'on n'ait à contrôler si les sorties sont bonnes, on doit l'entraîner en lui imposant un apprentissage de type supervisé. Rappelons le principe de l'apprentissage supervisé : un « professeur » donne un bon nombre de couples $(\overrightarrow{entrée}, \overrightarrow{sortie}_{désirée})$ pour que le réseau s'entraîne à associer à chaque entrée, la sortie correcte. A chaque stimulus d'information qu'on soumet au réseau, si la sortie obtenue n'est pas la sortie théorique du professeur, on modifie les coefficients de telle sorte que si on recommence, la sortie soit cette fois plus proche de la sortie désirée. La correction selon la règle de moindre carré permet de faire cela.

L'algorithme d'entraînement est la réitération de ce procédé sur de multiples exemples et de nombreuses fois. Il est possible de montrer que sous certaines conditions, les réponses du réseau sont sans fautes en un nombre fini d'itérations.

Règle du moindre carré pour un perceptron monocouche linéaire

Etudions le cas d'une couche $(N = 1)$ ayant une fonction de transfert linéaire $(f_1 = Id_{\mathbb{R}})$.

Pour commencer, on va s'intéresser à une couche ne contenant qu'un seul neurone $(s_1 = 1)$. Notre modélisation du réseau étant linéaire, on passera facilement au cas de plusieurs neurones en appliquant nos résultats à chacun d'entre eux (plus tard).

A chaque stimulus qu'on applique en entrée du réseau (sous la forme d'un vecteur $X \in \mathbb{R}^{e_1}$), on obtient une sortie $Y \in \mathbb{R}$ qui est *a priori* différente de la sortie désirée Y_d .On peut donc définir l'erreur pour chaque stimulus :

$$E = Y - Y_d \tag{3.29}$$

L'erreur quadratique étant le carré de cette erreur :

$$\epsilon = E^2 \tag{3.30}$$

La règle du moindre carré consiste à modifier les coefficients (poids et biais) de la couche de manière à réduire l'erreur quadratique.

Pour cela, on va « descendre le gradient » c'est-à-dire modifier les coefficients de W et b dans la direction de diminution de l'erreur quadratique. (figure 3.10)

Calcul du vecteur gradient de l'erreur quadratique $\nabla\epsilon$: en écrivant $W = [w_{ij}]$ et $X = [x_j]$ $(i = 1,\ 1 \leq j \leq e_1)$

$\forall\ 1 \leq j \leq e_1$,

$$\begin{aligned}[\nabla\epsilon]_j = \frac{\partial\epsilon}{\partial w_{1j}} = \frac{\partial E^2}{\partial w_{1j}} = 2E\frac{\partial E}{\partial w_{1j}} = 2E\frac{\partial(Y-Y_d)}{\partial w_{1j}} = 2E\frac{\partial((W\cdot X-b)-Y_d)}{\partial w_{1j}} \\ = 2E\frac{\partial(W\cdot X)}{\partial w_{1j}} = 2E\frac{\partial\sum_{k=1}^{e_1} w_{1k}x_k}{\partial w_{1j}} = 2x_jE\end{aligned} \tag{3.31}$$

Si bien que $\nabla\epsilon = 2EX$ et

$$\frac{\partial\epsilon}{\partial b} = 2E\frac{\partial((W\cdot X-b)-Y_d)}{\partial b} = -2E \tag{3.32}$$

Il devient donc très facile d'obtenir le gradient de l'erreur : il suffit de multiplier l'erreur par le stimulus d'entrée.

Pour diminuer l'erreur, la correction consiste à retrancher aux coefficients les gradients obtenus multipliés par une quantité positive η appelée *vitesse d'apprentissage*. On affecte ces nouvelles valeurs aux coefficients de la couche :

$$\begin{array}{rcl} W & \leftarrow & W - \eta\nabla\varepsilon = W - 2\eta E\ {}^tX \\ b & \leftarrow & b - \eta\frac{\partial\varepsilon}{\partial b} = b + 2\eta E \end{array} \tag{3.33}$$

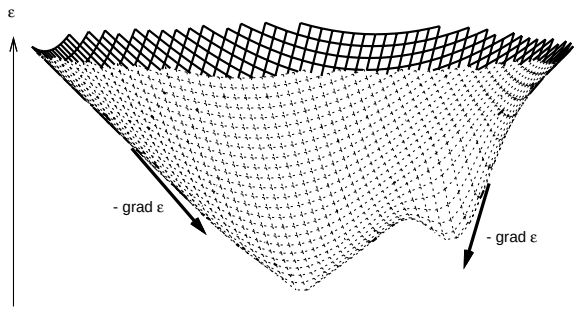


FIGURE 3.10 – La représentation de l'erreur quadratique en fonction des 2 poids d'un perceptron 2×1

Dans le cas plus général où on a un nombre s_1 de neurones dans notre couche, ce qui impose un biais $B \in \mathbb{R}^{s_1}$ et une matrice de poids $W \in M_{s_1,e_1}$, l'erreur quadratique est un réel positif qui s'exprime comme le carré de la norme du vecteur erreur $E = Y - Y_d$, soit dans $\mathbb{R}^{s_1}$:

$$\epsilon\ =\ {}^tE \cdot E \tag{3.34}$$

Pour chaque neurone i on modifie la composante du vecteur B et la ligne de la matrice W correspondantes de la même manière :

$$\begin{array}{ll} \forall 1 \leq i \leq s_1 & w_{ij} \leftarrow w_{ij} - 2\eta E_i x_j \quad (1 \leq j \leq e_1) \\ \forall 1 \leq i \leq s_1 & b_i \leftarrow b_i + 2\eta E_i \end{array} \tag{3.35}$$

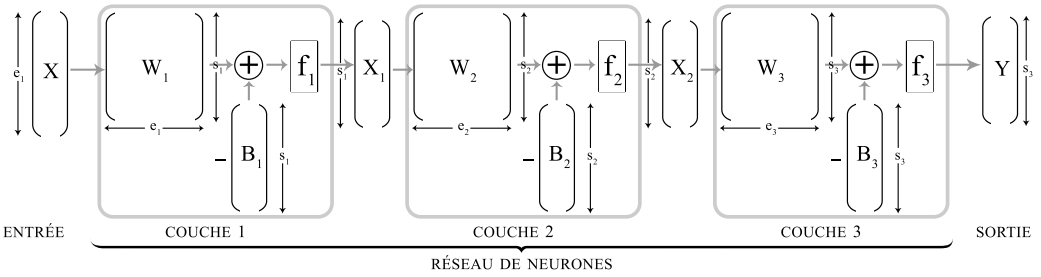
Soit sous forme matricielle :

$$\begin{array}{rcl} W & \leftarrow & W - 2\eta E^t X \\ B & \leftarrow & B + 2\eta E \end{array} \tag{3.36}$$

Rétropropagation pour un perceptron multicouche

Dans le cas d'un perceptron quelconque à N couches, on observe deux différences par rapport à l'étude précédente :

- Les fonctions d'activation de chaque couche ne sont pas forcément linéaires,
- Les données du « professeur » qui permettent la correction ne concernent que la sortie de la dernière couche (couche de sortie).



Il va alors falloir trouver un moyen de corriger les couches une par une en partant de la dernière et en remontant pas à pas jusqu'à la première. C'est en quoi consiste la méthode de rétropropagation du gradient.

L'algorithme de correction est toujours basé sur la descente du gradient, par exemple sur la couche de sortie N :

$$\begin{array}{rcl} W_N \leftarrow & W_N - \eta \nabla \varepsilon_N \\ B_N \leftarrow & B_N - \eta \nabla' \varepsilon_N \end{array} \tag{3.37}$$

Mais ici les dérivées partielles ont changé car la fonction d'activation n'est pas toujours linéaire.

Cependant, si les fonctions d'activation sont dérivables (ce qui n'est pas le cas de la fonction seuil : sa dérivée est nulle partout sauf en 0 où elle n'est pas définie), et en notant f'_n la dérivée de f_n, on peut montrer qu'il faut corriger les poids et biais de la sorte :

$$\forall 1 \leq n \leq N, \quad \begin{array}{rcl} W_n & \leftarrow & W_n - \eta S_n {}^t X_n \\ B_n & \leftarrow & B_n + \eta S_n \end{array} \tag{3.38}$$

Où S_n est un vecteur qui représente la sensibilité de l'erreur au changement de $A_n = [a_i^{(n)}] = W_n \cdot X_{n-1} - B_n$ dans la couche n et se calcule par rétro-récurrence :

$$S_N = 2 \begin{pmatrix} f'_N\left(a_1^{(N)}\right) & 0 & \cdots & 0 \\ 0 & f'_N\left(a_2^{(N)}\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f'_N\left(a_{s_N}^{(N)}\right) \end{pmatrix} \cdot (Y - Y_d) \tag{3.39}$$

$$\forall 1 \leq n \leq N-1, \; S_n = \begin{pmatrix} f'_n\left(a_1^{(n)}\right) & 0 & \cdots & 0 \\ 0 & f'_n\left(a_2^{(n)}\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f'_n\left(a_{s_n}^{(n)}\right) \end{pmatrix} \cdot {}^t W_{n+1} \cdot S_{n+1} \tag{3.40}$$

L'algorithme d'apprentissage par rétropropagation du gradient peut être résumé par les étapes :

– On propage le vecteur d'entrée X à travers les couches, ce qui donne $X_1, \ldots, X_{N-1}, Y$,
– On calcule la dernière sensibilité S_N ce qui nous permet de corriger les poids et biais de la dernière couche ; puis on calcule S_{N-1} pour corriger l'avant dernière, ..., et ce jusqu'à la première couche par rétropropagation.
Etapes que l'on répète pour chaque couple entrée/sortie désirée (X, Y_d) donnée par le professeur. Chaque itération (correspondant à une certaine entrée) modifie les coefficients du réseau d'une manière qui est censée diminuer l'erreur quadratique pour l'entrée donnée. Pour arriver à un résultat, il convient d'entraîner notre réseau plusieurs fois pour chaque couple entrée/sortie désirée, pour arriver à une erreur quadratique moyenne la plus basse possible.
Remarque : il existe d'autres méthodes de correction de l'erreur, telle la méthode de Newton ou celle du gradient conjugué qui consiste à chercher le minimum de l'erreur selon des droites dans l'espace des coefficients.

3.2.8 Conclusion

Un réseau de neurones artificiels est un modèle de calcul dont la conception est très schématiquement inspirée du fonctionnement des neurones biologiques.

Chapitre 4

Simulation en Matlab du système de reconnaissance réalisé

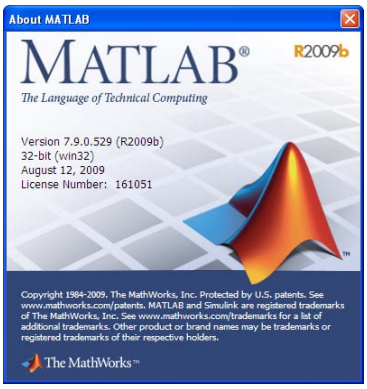
4.1 Introduction

Nous allons présenter dans ce chapitre les expériences réalisées durant la simulation du système de reconnaissance réalisé, ainsi que les résultats et les interprétations. Mais d'abord, nous allons décrire brièvement notre système. La simulation a été faite avec le logiciel de calcul matriciel Matlab (version R2009b).

Le système utilise l'algorithme PCA pour la reconnaissance. Il va faire une mesure de similarité après l'extraction des caractéristiques du visage de test à partir de son image présentée à l'entrée.

D'abord, le système doit identifier la personne, c.-à-d. que le système doit décider laquelle des images stockées dans la base de données ressemble plus à l'image à identifier. Puis, il doit vérifier si elle appartient à la base de données ou non. En d'autres termes, le système doit décider si la personne est un client ou non.

La mesure de similarité peut se faire de différentes manières (distance euclidienne, la mesure en angle, la mesure de corrélation). Dans notre projet, nous avons utilisé la distance euclidienne



pour cette mesure. Ensuite, nous avons essayé d'améliorer les résultats de la PCA en la combinant avec un réseau de neurones multicouche. Ce dernier remplacera la distance euclidienne dans l'identification de la personne en trouvant la classe de son vecteur caractéristique (poids).

Nous avons essayé aussi d'améliorer la PCA par :

- L'utilisation d'une base de données qui contient plus d'une image pour chaque client.
- Augmenter le nombre des visages propres gardés.

4.2 Base de données

Nos expériences ont été faites sur des images frontales de visages tirées de la base de données obtenue de la référence [23]. Vous trouverez une description de cette base dans l'annexe B. Les images sont en couleurs, de résolution (200×180 pixels) et de format jpg.

Il n'est pas suffisant d'utiliser la même base de données pour pouvoir honnêtement comparer des résultats. Il est nécessaire également de définir un protocole de test. Dans le protocole de Lausanne [24], la base de données est scindée en trois ensembles : ensemble d'apprentissage, ensemble de validation et ensemble de test. L'ensemble d'apprentissage est utilisé comme ensemble de référence. Il sert d'ensemble de base (dorénavant, nous l'appellerons base tout court). L'ensemble d'évaluation permet de fixer les paramètres du système de reconnaissance de visages. Enfin, l'ensemble de test permet de tester le système en lui présentant des images de personnes lui étant totalement inconnues.

En fait, la base de données est divisée en deux classes : clients est imposteurs. L'ensemble d'apprentissage ne contient que des clients. Les imposteurs sont répartis dans les deux autres ensembles.

Pour les besoins de l'implémentation, nous étions ramenés à construire 3 bases de données différentes en terme du nombre de personnes et de la résolution des images. Ces bases sont définies comme suit :

Première base :

Ensemble	Clients	Imposteurs
Apprentissage	150	0
Evaluation	600 (4 images par personne)	160 (8 images par personne)
Test	450 (3 images par personne)	450 (9 images par personne)

Deuxième base :

Ensemble	Clients	Imposteurs
Apprentissage	40	0
Evaluation	160 (4 images par personne)	100 (5 images par personne)
Test	120 (3 images par personne)	120 (6 images par personne)

Troisième base :

Cette base est identique à la précédente sauf que la résolution des images prises est réduite à $(25 \times 23$ pixels) en faisant une compression des images originales. En fait, la première base est celle avec laquelle nous avons fait les premières expériences. Mais, plus tard, nous avons créé les deux autres pour les utilisées dans l'implémentation.

4.3 Mesure de la performance

Si chaque client doit être accepté et chaque imposteur rejeté, on obtiendra les 4 taux de reconnaissance suivants :

- TFR
- TFA
- TBA
- TBR

Les TFR et TFA sont respectivement le Taux de Faux Rejet et le Taux de Fausse Acceptation. Pour un bon système, ils doivent être les plus faibles possibles. Les TBA et TBR sont respectivement le Taux de Bonne Acceptation et le Taux de Bon Rejet. Ces taux caractérisent ce que l'on appelle en reconnaissance de visages le pouvoir d'identification.

Il est important de noter que le TFA peut se présenter selon deux cas de figure ; soit que le système accepte un imposteur ou encore il accepte un autre client.

En supposant que le système d'identification soit strict. Le TFA sera faible mais le TFR sera fort élevé. Au contraire, un système laxiste sera caractérisé par un TFA élevé et un TFR plutôt bas. Le juste milieu se situe quelque part entre les deux, il se trouvera au Taux d'Egale Erreur ou TEE illustré par la figure 4.1.

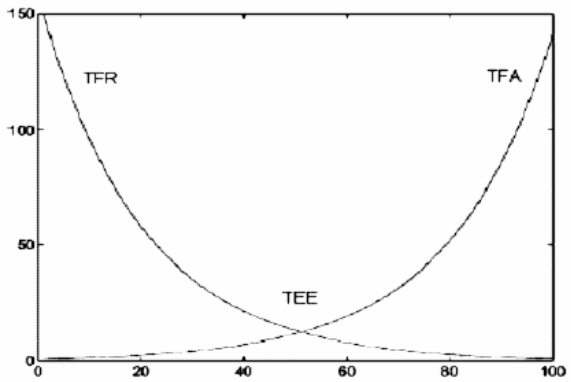


FIGURE 4.1 – TFA, TFR et TEE en fonction du laxisme du système

Les deux taux d'erreurs (TFA, TFR) sont calculés dans les deux ensembles : D'abord, dans l'ensemble d'évaluation, qui va permettre de fixer le TEE en faisant varier le seuil d'acceptation d'une personne, ce seuil va déterminer le minimum de ressemblance entre cette personne et la base, pour admettre qu'elle appartient à cette base.

Ensuite, ces taux vont être calculés dans l'ensemble de test, en utilisant le seuil fixé précédemment pour vérifier la robustesse du système.

4.4 Travail effectué

4.4.1 Première base (150 personnes)

Nous allons présenter ici les résultats obtenus avec la première base composée de 150 personnes. Nous commencerons par la PCA seule et ensuite, nous ajouterons le réseau de neurones multicouche.

1) PCA

Nous verrons la constitution de la base et les imposteurs utilisés dans les deux ensembles d'évaluation et de test. Chaque personne est représentée par une de ses images.

La base (d'apprentissage) :

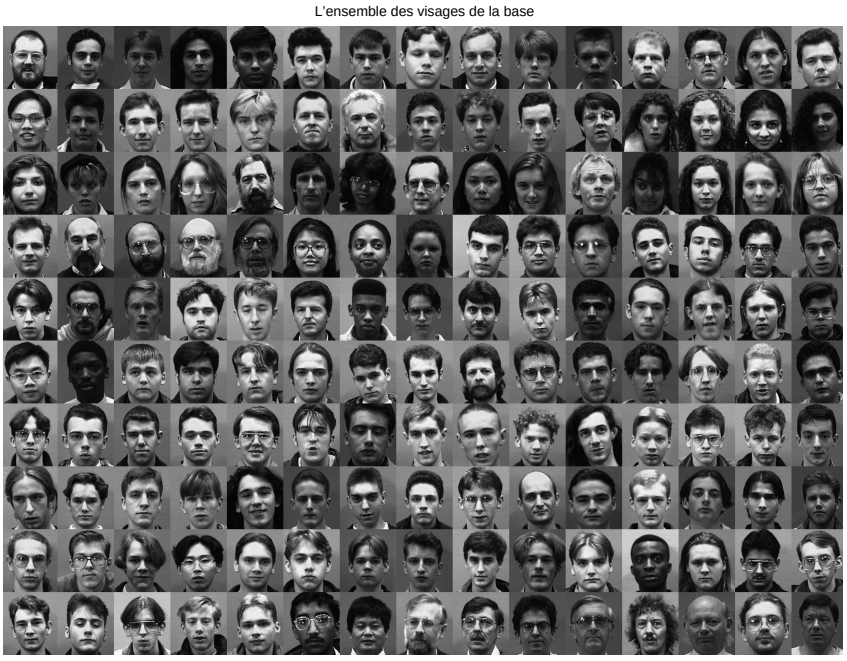


FIGURE 4.2 – L'ensemble des images de la base (150)

Les imposteurs :
Les imposteurs utilisés dans l'ensemble d'évaluation :

FIGURE 4.3 – Les imposteurs de la base d'évaluation

Les imposteurs utilisés dans l'ensemble de test :

FIGURE 4.4 – Les imposteurs de la base de test

Pour les tests de performances, nous avons fait trois expériences :
- Dans la première expérience, nous avons augmenté le nombre de visages propres gardés.
- Dans la deuxième expérience, nous avons utilisé plusieurs images pour chaque client dans la base tout en gardant un même pourcentage des visages propres à utiliser (6.6667% du nombre total d'images dans la base).
- Dans la dernière expérience, nous avons utilisé plusieurs images pour chaque client dans la base tout en gardant cette fois le même nombre de visages propres à utiliser (à savoir 10 visages propres).

Nous nous intéresserons uniquement aux taux d'erreur TFA et TFR et à leur somme qui caractérisera la performance du système.

Nous donnerons le temps de réponse du système que nous appellerons temps de calcul. Ce temps représente le temps que prend le système pour identifier une personne.

Soit k le nombre de visages propres et *vbase* le nombre d'images de chaque client dans la base.

1re expérience : Fixer *vbase*=1 et varier k

Voici dans la figure ci-dessous la totalité des visages propres de cette base. Cette figure contient 150 visages propres.

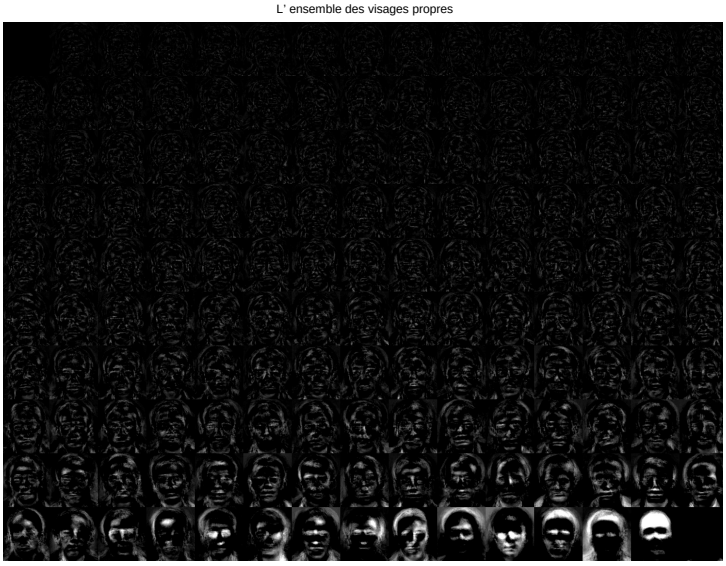


FIGURE 4.5 – L'ensemble des visages propres

On remarque que les visages propres sont classés du moins significatif au plus significatif, du haut vers le bas et de la gauche vers la droite. C'est Matlab qui fait automatiquement la classification des vecteurs propres par ordre croissant de leurs valeurs propres.

On voit bien que les derniers visages propres (les plus significatifs et qui ont les valeurs propres les plus grandes) portent plus de motifs (informations) que les autres qui sont presque noirs. Ce qui nous amène, à faire le choix des visages propres à garder parmi ces derniers.

Voici les résultats obtenus dans cette expérience :

Ensemble d'évaluation : (Clients : 150×4/ Imposteurs : 20×8)

Taux \ k	5	7	10	15	20	25	30
TFA(%)	13.9474	11.0526	10.0000	9.7368	9.8684	9.6053	9.34216
TFR(%)	13.9474	12.5000	10.0000	10.7895	10.9211	11.0526	11.7105
(TFA+TFR) (%)	27.8948	23.5526	20.0000	20.5263	20.7895	20.6579	21.0526

TABLE 4.1 – Variation des taux en fonction de k pour l'ensemble d'évaluation

Ensemble de test : (Clients : 150×3/ Imposteurs : 50×9)

Taux \ k	5	7	10	15	20	25	30
TFA(%)	15.7778	12.1111	21.6667	23.1111	21.6667	20.1111	18.1111
TFR(%)	8.0000	7.2222	5.1111	5.7778	5.6667	5.4444	6.2222
(TFA+TFR) (%)	23.7778	19.3333	26.7778	28.8889	27.3334	25.5555	24.3333

TABLE 4.2 – Variation des taux en fonction de k pour l'ensemble de test

Temps de calcul :

k	5	7	10	15	20	25	30
Temps de calcul (sec)	12.3420	12.4080	12.5700	12.5400	12.7620	14.6100	26.2440

TABLE 4.3 – Variation du temps de calcul en fonction de k

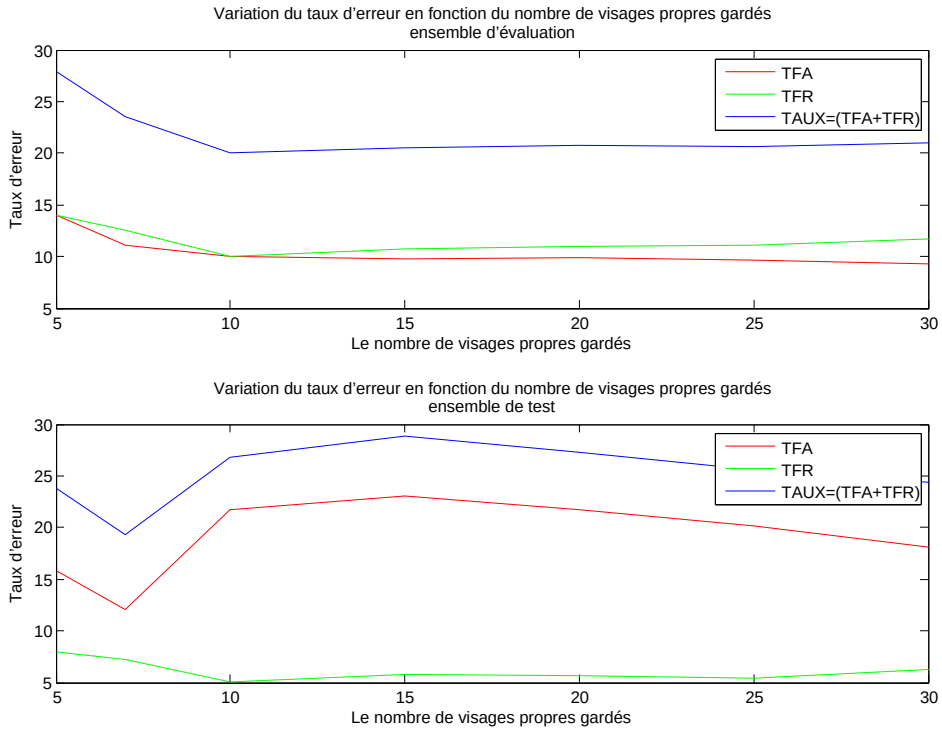


FIGURE 4.6 – Variation du taux d'erreur en fonction du nombre des visages propres gardés

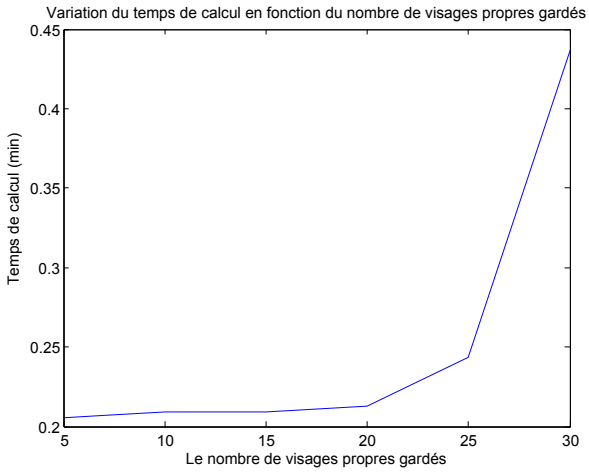


FIGURE 4.7 – Variation du temps de calcul en fonction du nombre de visages propres gardés

Interprétations :

Ensemble d'évaluation :

– On remarque que l'augmentation du nombre de visages propres minimise le taux d'erreur global jusqu' à une certaine valeur.
Cette valeur représentera alors la limite des visages propres où se trouve la majorité d'informations contenues dans la base.

– On remarque aussi que TFA et TFR sont presque identiques car on a ajusté le seuil de sorte que l'on ait TFA=TFR=TEE (Taux d'Egale Erreur).
Dans toutes les expériences qui vont suivre, nous avons fixé le seuil de manière à obtenir TFA=TFR= TEE.

Ensemble de test :

– On remarque que le TFR a beaucoup diminué contrairement au TFA qui a augmenté. Aussi, on remarque que le taux d'erreur global n'est plus constant.
Tous ces résultats sont attendus, à cause du nombre élevé des imposteurs qui se trouvent dans cet ensemble (450 images).

Temps de calcul :

– Pour le temps de calcul, on remarque qu'il varie proportionnellement avec le nombre de visages propres gardés. Ce qui est tout à fait normal, puisqu'on ajoute au programme des calculs supplémentaires. On remarque également, que les temps ne dépassent pas les 30 secondes, ce qui est acceptable.

2e expérience : Varier les deux : *vbase* et k

Nous avons varié cette fois-ci *vbase* de 1 à 5 en gardant le même pourcentage des vecteurs propres à chaque fois tel que $k = 10 \cdot vbase$.

Voici les résultats obtenus :

Ensemble d'évaluation : (Clients : 150×4/ Imposteurs : 20×8)

vbase / k, Taux	1	2	3	4	5
k	10	20	30	40	50
TFA(%)	10.0000	2.1053	1.1842	1.0526	0.7895
TFR(%)	10.0000	2.2368	1.0526	1.0526	0.7895
(TFA+TFR) (%)	20.0000	4.3421	2.2368	2.1052	1.5790

TABLE 4.4 – Variation de k et des taux en fonction de *vbase* pour l'ensemble d'évaluation

Ensemble de test : (Clients : 150×3/ Imposteurs : 50×9)

vbase / k, Taux	1	2	3	4	5
k	10	20	30	40	50
TFA(%)	21.6667	3.6667	0.4444	0.4444	0.1111
TFR(%)	5.1111	2.6667	1.6667	1.1111	0.6667
(TFA+TFR) (%)	26.7778	6.3334	2.1111	1.5555	0.7778

TABLE 4.5 – Variation de k et des taux en fonction de *vbase* pour l'ensemble de test

Temps de calcul :

vbase	1	2	3	4	5
Temps de calcul (min)	0.2257	0.8029	1.7416	3.5183	7.3665

TABLE 4.6 – Variation du temps de calcul en fonction de *vbase*

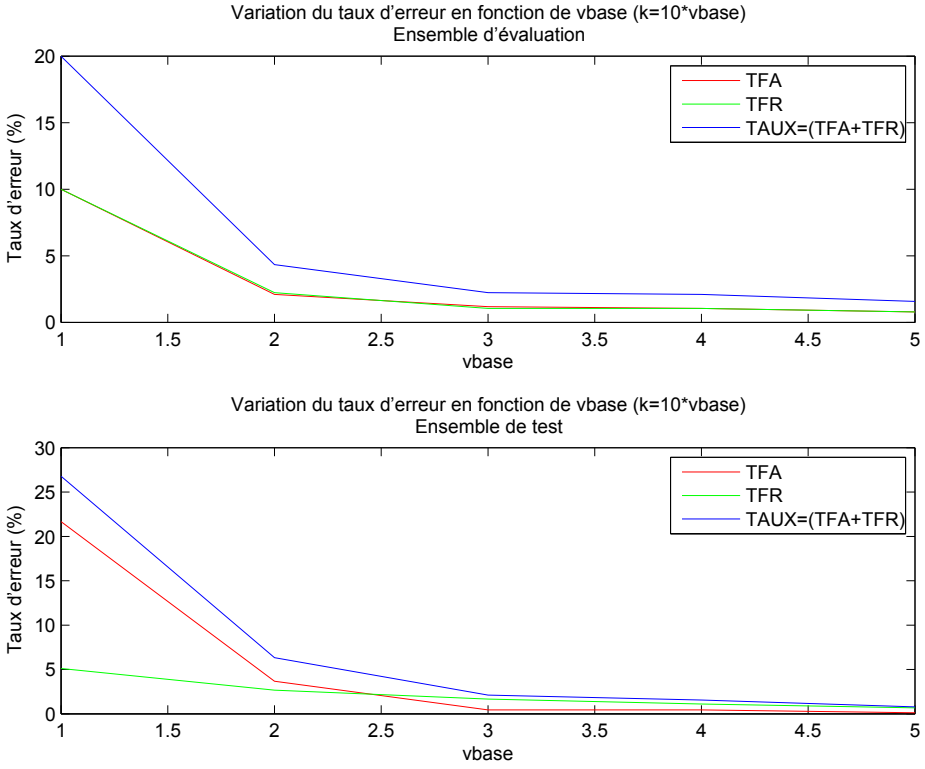


FIGURE 4.8 – Variation du taux d'erreur en fonction de *vbase* ($k = 10 \cdot vbase$)

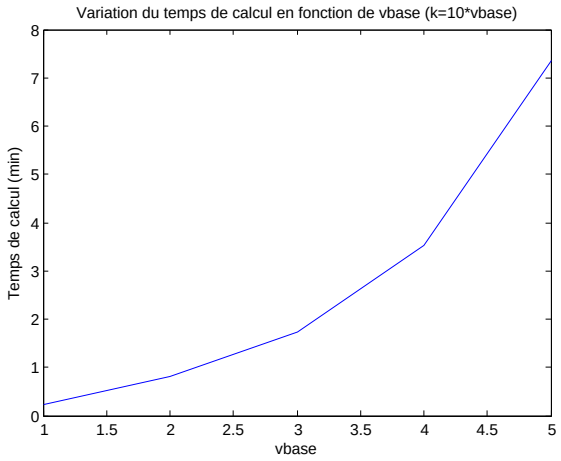


FIGURE 4.9 – Variation du temps de calcul en fonction de *vbase* ($k = 10 \cdot vbase$)

Interprétations :

Ensembles :

– On remarque que l'utilisation de plus d'une image pour les clients dans la base, tout en ne gardant que 6.6667% des visages propres ($k = 10 \cdot vbase$) chaque fois, a fait diminuer l'erreur globale et ainsi augmenter la performance du système.
Dès qu'on arrive à 3 images par personne dans la base, le taux d'erreur descend au dessous de 5%, mais 2 images par personne donnent déjà une diminution considérable de ce taux (de 20-27% à 4-6%) ce qui peut être satisfaisant pour des systèmes peu stricts.

Temps de calcul :

– On remarque sa variation proportionnelle avec le nombre d'images de chaque personne dans la base, ce qui est tout à fait normal. Il est à noter aussi la gamme de variation de ce temps. Deux images et moins pour chaque personne, donnent un temps de calcul inférieur à 1 minute mais dès qu'on dépasse 2 images, le temps de calcul devient considérable et dépasse les 7 minutes avec 5 images par personne.

3e expérience : k fixe=10 et *vbase* variable

Dans cette dernière expérience, nous avons varié *vbase* de 1 à 5 tout en gardant k fixe, la valeur choisie est 10. Voici les 10 visages propres gardés :

FIGURE 4.10 – L'ensemble des visages propres gardés

Les visages ressemblent plus à des visages (**fantômes**) qu'à des visages normaux.

Voici les résultats obtenus :

Ensemble d'évaluation : (Clients : 150×4/ Imposteurs : 20×8)

Taux \ *vbase*	1	2	3	4	5
TFA(%)	10.0000	2.7632	1.7105	1.3158	1.3158
TFR(%)	10.0000	2.8947	1.8421	1.4474	1.3158
(TFA+TFR) (%)	20.0000	5.6579	3.5526	2.7632	2.6316

TABLE 4.7 – Variation des taux en fonction de *vbase* pour l'ensemble d'évaluation

Ensemble de test : (Clients : 150×3/ Imposteurs : 50×9)

Taux \ *vbase*	1	2	3	4	5
TFA(%)	21.6667	3.3333	2.6667	2.5556	1.6667
TFR(%)	5.1111	3.1111	2.4444	2.0000	1.6667
(TFA+TFR) (%)	26.7778	6.4444	5.1111	4.5556	3.3334

TABLE 4.8 – Variation des taux en fonction de *vbase* pour l'ensemble de test

Temps de calcul :

vbase	1	2	3	4	5
Temps de calcul (min)	0.7779	1.8433	2.9919	4.0655	5.8470

TABLE 4.9 – Variation du temps de calcul en fonction de *vbase*

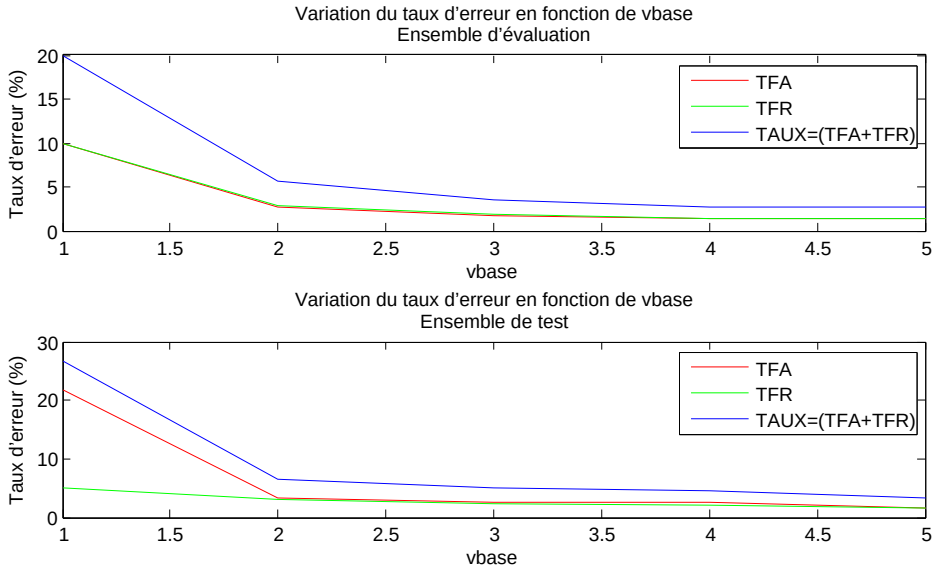


FIGURE 4.11 – Variation du taux d'erreur en fonction de *vbase*

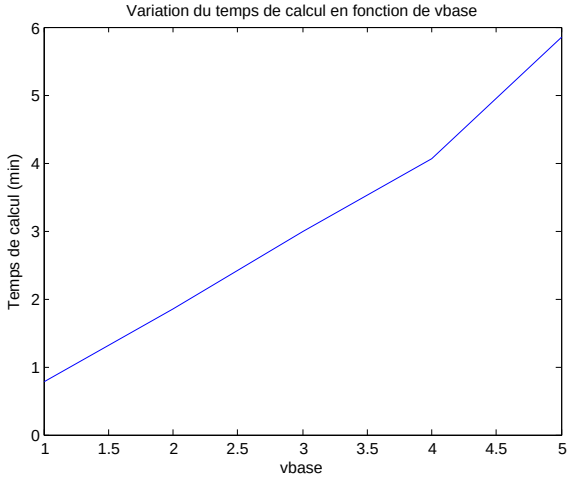


FIGURE 4.12 – Variation du temps de calcul en fonction de *vbase*

Interprétations :

Ensembles :

– On remarque qu'il n'y a pas une grande différence entre cette expérience et celle qui la précède concernant les résultats des deux ensembles d'évaluation et de test. Donc, on conclue qu'il suffit de décider d'un certain nombre de visages propres k et de l'utiliser quelque soit le nombre d'images par personne à utiliser dans la base. Il est à noter seulement que la décision sur ce nombre peut être faite à travers la première expérience.

Temps de calcul :

– Il augmente toujours proportionnellement à *vbase*, mais cette fois d'une façon presque linéaire. On remarque aussi qu'il n'atteint pas les 6 minutes alors qu'il dépassait les 7 minutes dans la 2^{e} expérience.
– On remarque aussi que les TFA et TFR sont égaux hormis pour $vbase = 1$ dans l'ensemble de test, ce qui peut être interprété par un bon choix des seuils.

2) PCA_RNA

Nous avons essayé dans cette section d'améliorer le taux de reconnaissance en remplaçant la distance euclidienne par un réseau de neurones multicouche pour faire l'identification.

Les expériences de la section précédente étant faites, nous avons fixé 10 visages propres à garder et une image par personne dans la base.

Pour l'apprentissage, nous avons fait beaucoup d'expériences afin d'arriver à un réseau convenable et performant. Ces expériences nous ont amené à jouer sur plusieurs paramètres que nous exposerons dans la sous-section suivante.

Après la présentation du réseau réalisé, nous avons évalué ses performances afin de les comparer avec les performances obtenues par utilisation de la distance euclidienne. Nous avons considéré pour cela, les résultats de la 3^{e}expérience de la section précédente.

Préparation et ajustement du réseau

1. Algorithmes d'entrainement : Matlab propose (Neural Network Toolbox) plusieurs types d'algorithmes qui permettent un entrainement rapide du réseau. Mais, chaque algorithme a ses avantages et ses inconvénients. On peut donner l'exemple de l'algorithme **trainrp** qui est le plus rapide sur les problèmes de reconnaissance de formes. Cependant, il n'est pas performant sur les problèmes d'approximation des fonctions. [25]

   Parmi les algorithmes testés dans notre projet, on cite : **trainrp** et **trainlm**.

   L'algorithme gardé par la suite est : **trainrp** vu sa rapidité.

2. Le nombre d'itérations : le nombre d'itérations que Matlab utilise par défaut durant l'apprentissage est 1000 itérations, cependant si on l'atteint, l'entraînement du réseau s'arrête malgré que la sortie désirée ne soit pas atteinte. Mais, on peut le changer comme on veut en utilisant la commande : nom du réseau.**trainParam.epochs**=nombre d'itérations voulu.

   Exemple : net.trainParam.epochs=2000.

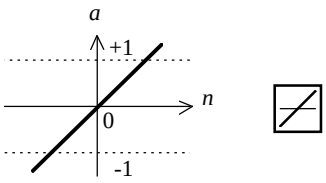
   Ce nombre d'itérations représente le maximum d'itérations à faire. Donc, l'apprentissage peut s'arrêter avant, quand la sortie désirée est atteinte.

3. Erreur quadratique : L'erreur quadratique vaut 0 par défaut, mais on peut aussi la changer si l'apprentissage prend beaucoup de temps en essayant de l'atteindre, et dans ce cas, on doit tolérer la baisse de la précision des résultats. la commande qui permet de changer l'erreur est : nom du réseau.**trainParam.goal**=erreur voulue.

   Exemple : net.trainParam.goal=0.00001.

4. Le nombre des couches cachées et le nombre de neurones dans chaque couche.

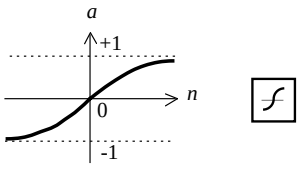
5. La fonction d'activation : nous avons essayé avec trois fonctions de transfert qui sont :
   - La fonction de transfert linéaire, notation Matlab : **purelin**



a = purelin(n)

Linear Transfer Function

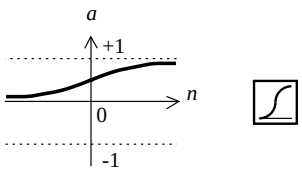
   - La tangente hyperbolique, notation Matlab : **tansig**



a = tansig(n)

Tan-Sigmoid Transfer Function

– La sigmoïde, notation Matlab : **logsig**



a = logsig(n)

Log-Sigmoid Transfer Function

Les deux premières fonctions d'activation n'ont pas donné de bons résultats, c'est la fonction **logsig** qui a donnée les meilleurs résultats pour notre application.

6. Le nombre de sorties : D'abord, nous avons essayé d'avoir une seule sortie décimale qui donne l'indice de la personne identifiée qui sera entre 1 et 150 (150 personnes), et pour faire cela, nous avons utilisé la fonction d'activation **purelin**. Mais le réseau ainsi réalisé a donné de mauvais résultats.

   Ensuite, nous avons décidé de coder les 150 indices sur 8 bits (binaires) où chaque bit est représenté par une sortie. Ce réseau a donné de meilleurs résultats mais il reste insuffisant.

   Enfin, nous avons choisi 150 sorties telles que le client d'indice **i** positionne la sortie **i** à 1 et les autres sorties à 0.

Le réseau réalisé

Le réseau réalisé contient dix entrées (puisqu'on a gardé 10 vecteurs propres, donc 10 poids), une seule couche cachée composée de 256 neurones et une couche de sortie contenant 150 neurones. La fonction d'activation utilisée dans ces deux couches est la même, à savoir la fonction **logsig** et l'algorithme d'apprentissage utilisé est le **trainrp**. L'apprentissage a pris 65 itérations.

Voici ce que donne Matlab lors de l'apprentissage :

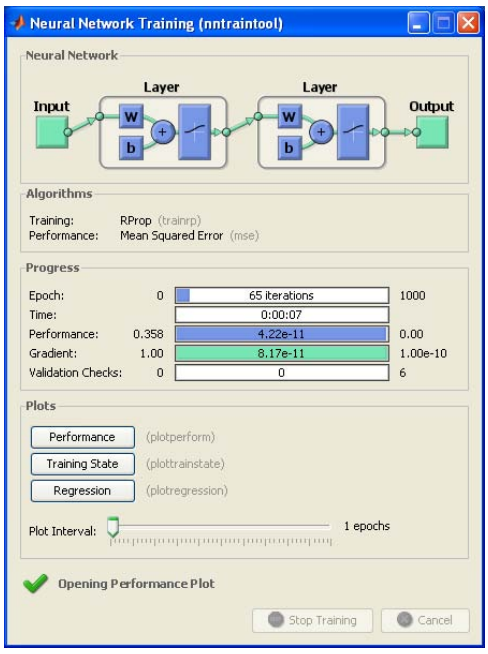


FIGURE 4.13 – Apprentissage du réseau

Le bouton Performance dans cette fenêtre donne la variation de l'erreur quadratique en fonction du nombre d'itérations.

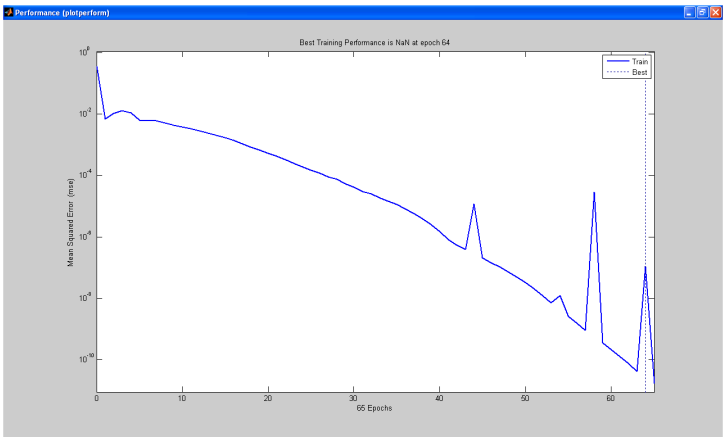


FIGURE 4.14 – Variation de l'erreur quadratique en fonction du nombre d'itérations

Evaluation et comparaison des performances du réseau

Nous avons fait ici 3 comparaisons. Les deux premières avec les ensembles d'évaluation et de test et la dernière avec un ensemble ne contenant que les clients de l'ensemble d'évaluation où nous faisons varier le nombre d'images par client. En fait, nous avons enlevé les imposteurs afin d'éviter une éventuelle divergence du réseau de neurones.

– **L'ensemble d'évaluation** :

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	10.0000	11.5789
TFR(%)	10.0000	10.0000
(TFA+TFR) (%)	20.0000	21.5789

TABLE 4.10 – Comparaison des taux de l'ensemble d'évaluation entre PCA et PCA_RNA

– **L'ensemble de test** :

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	21.6667	22.7778
TFR(%)	5.1111	5.1111
(TFA+TFR) (%)	26.7778	27.8889

TABLE 4.11 – Comparaison des taux de l'ensemble de test entre PCA et PCA_RNA

– **Le temps de calcul** :

Distance euclidienne	RNA
0.7779 min	0.2738 min

TABLE 4.12 – Comparaison du temps de calcul entre PCA et PCA_RNA

– **3e ensemble** : (Clients de l'ensemble d'évaluation)

Méthode / Nbre d'images par client	Distance euclidienne (Taux (%))	RNA (Taux(%))
1	12.6667	14
2	14	16
3	14	16.2222
4	13.8333	17

TABLE 4.13 – Comparaison des taux du 3e ensemble entre PCA et PCA_RNA

Interprétations :

– **Les ensembles d'évaluation et de test** :
Malheureusement, le réseau de neurones n'a pas donné de meilleurs résultats par rapport à la distance euclidienne. Néanmoins, la différence entre les taux d'erreur n'est pas très grande.
Pour l'ensemble d'évaluation, elle est de 1.5789% (12 personnes) (TFR reste toujours le même puisqu'on utilise le même seuil pour l'authentification).
Pour l'ensemble de test, nous avons eu une différence de 1.1111% (10 personnes).

– **Le temps de calcul** :
Le temps de calcul du réseau de neurones est inférieur à celui de la distance euclidienne. Ce qui constitue un point positif pour le réseau de neurones.

– **Dernier ensemble** :
On remarque que la distance euclidienne donne toujours les meilleurs résultats. - Le taux donné ne peut être que le TFA puisqu'on n'utilise que des images de clients.

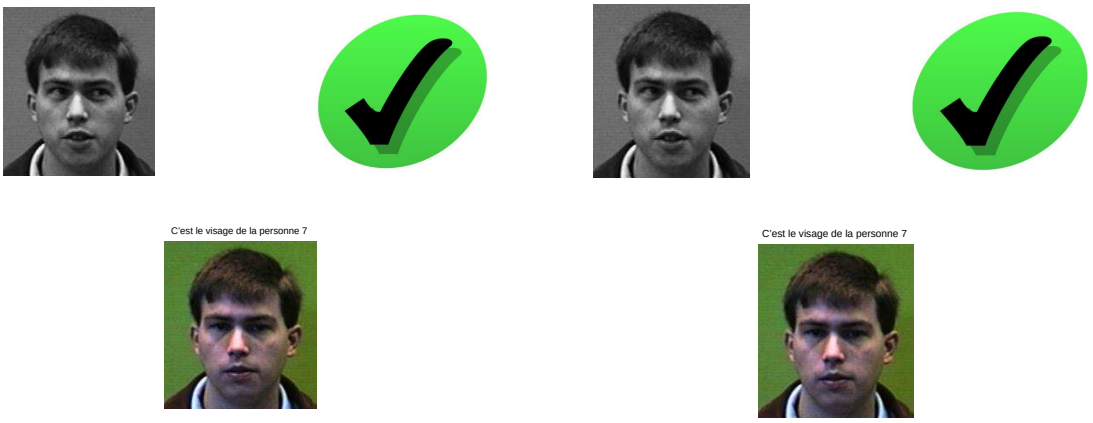
3) Présentation de quelques résultats

Nous allons présenter des exemples des différents cas de reconnaissance que donne notre système avec la distance euclidienne et le réseau de neurones.

Il est à noter qu'il y a un cas que nous n'avons pas traité dans notre projet. Ce cas est la présence d'une ambigüité dans la décision de telle sorte que le système indique que la personne à reconnaître se trouve à égale distance de deux personnes ou plus, et il ne peut pas faire une décision sur la bonne personne.

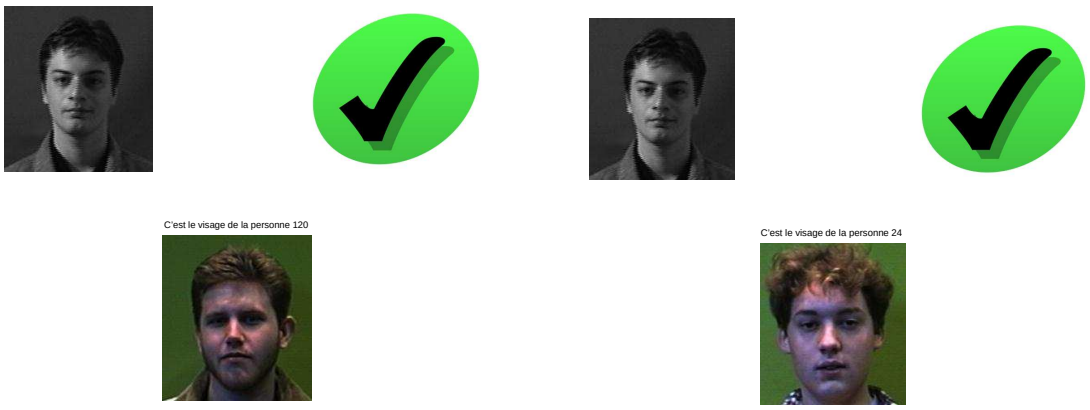
La cause de notre omission de ce cas est la précision élevée avec laquelle travaille Matlab ainsi que le nombre d'images existantes dans notre base qui n'est pas vraiment élevé, donc ce cas ne se présentera pas.

Les figures de 4.15 à 4.19 donnent des exemples pour les différents cas de reconnaissance qui se présentent :



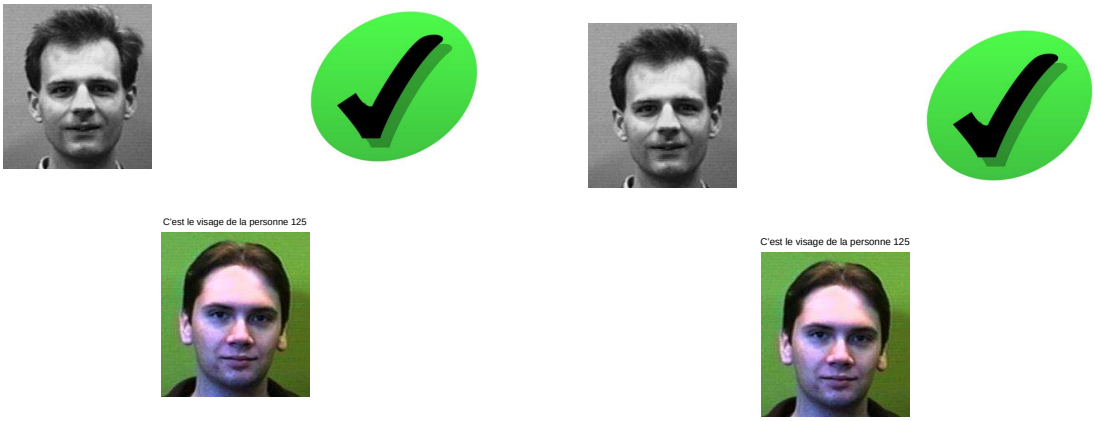
(a) Distance euclidienne (b) Réseau de neurones

FIGURE 4.15 – Bonne acceptation



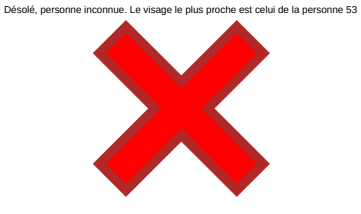
(a) Distance euclidienne (b) Réseau de neurones

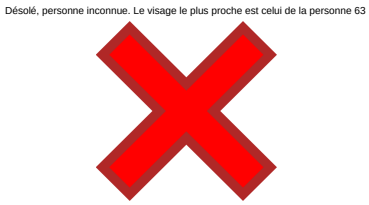
FIGURE 4.16 – Fausse acceptation (Imposteur accepté)



(a) Distance euclidienne (b) Réseau de neurones

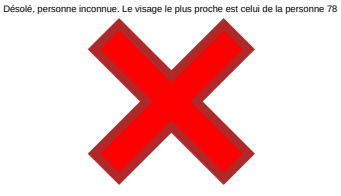
FIGURE 4.17 – Fausse acceptation (Client mal identifié)





(a) Distance euclidienne (b) Réseau de neurones

FIGURE 4.18 – Bon rejet



(a) Distance euclidienne (b) Réseau de neurones

FIGURE 4.19 – Faux rejet

Remarque :
On remarque que les résultats d'identification sont identiques sauf que le réseau de neurones ne donne pas forcément les mêmes indices pour la personne la plus proche.

4.4.2 Deuxième base (40 personnes)

Nous allons présenter ici les résultats obtenus avec la première base composée de 40 personnes. Nous commencerons par la PCA seule et ensuite, nous ajouterons le réseau de neurones multicouche comme nous l'avons fait pour la première base.

1) PCA

Nous verrons la constitution de la base et les imposteurs utilisés dans les deux ensembles d'évaluation et de test. Chaque personne est représentée par une de ses images.

La base (d'apprentissage) :

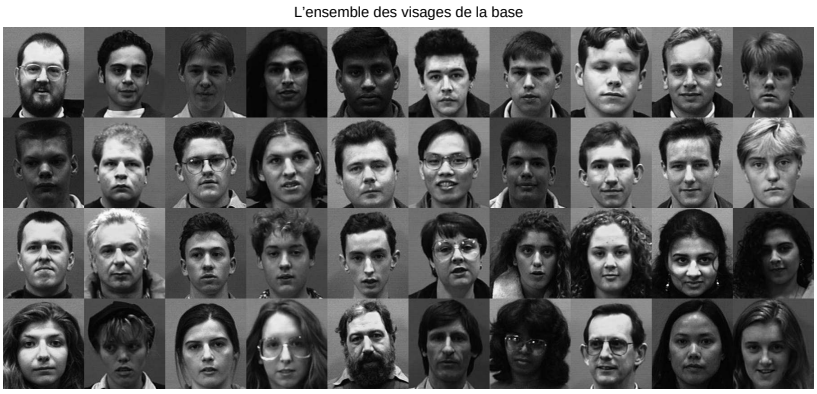


FIGURE 4.20 – L'ensemble des images de base (40)

Les imposteurs :
Les imposteurs utilisés dans l'ensemble d'évaluation :

FIGURE 4.21 – Les imposteurs de la base d'évaluation

Les imposteurs utilisés dans l'ensemble de test :

FIGURE 4.22 – Les imposteurs de la base de test

Cette fois, nous avons fait une seule expérience où nous avons fait varier *vbase* de 1 à 5, tout en gardant 5 visages propres (12.5%). De même, nous fixons les paramètres avec l'ensemble d'évaluation et nous testons les performances avec l'ensemble de test.

Voici l'ensemble des visages propres gardés :

FIGURE 4.23 – L'ensemble des visages propres gardés

Les résultats obtenus sont les suivants :
Ensemble d'évaluation : (Clients : 40×4/ Imposteurs : 20×5)

Taux \ *vbase*	1	2	3	4	5
TFA(%)	11.9231	4.6154	3.0769	2.6923	2.6923
TFR(%)	11.9231	4.2308	3.0769	2.6923	2.6923
(TFA+TFR) (%)	23.8462	8.8462	6.1538	5.3846	5.3846

TABLE 4.14 – Variation des taux en fonction de *vbase* pour l'ensemble d'évaluation

Ensemble de test : (Clients : 40×3/ Imposteurs : 20×6)

	1	2	3	4	5
TFA(%)	6.6667	0.8333	0.4167	0	0
TFR(%)	8.7500	5.0000	3.7500	3.3333	2.9167
(TFA+TFR) (%)	15.4167	5.8333	4.1667	3.3333	2.9167

TABLE 4.15 – Variation des taux en fonction de *vbase* pour l'ensemble de test

Temps de calcul :

vbase	1	2	3	4	5
Temps de calcul (min)	0.0971	0.1078	0.1710	0.2742	0.4132

TABLE 4.16 – Variation du temps de calcul en fonction de *vbase*

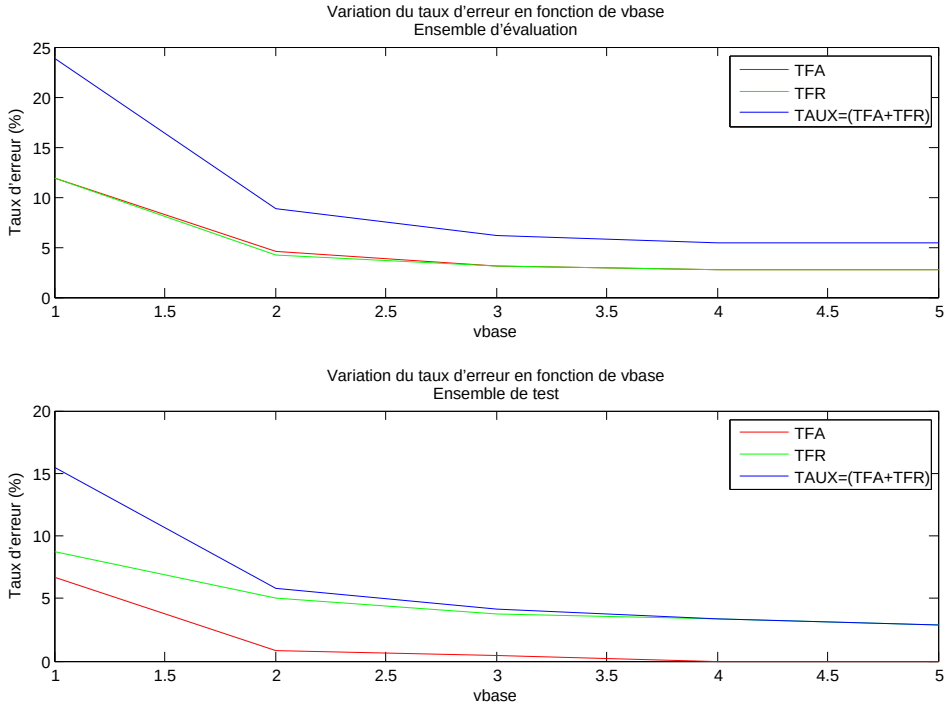


FIGURE 4.24 – Variation du taux d'erreur en fonction de *vbase*

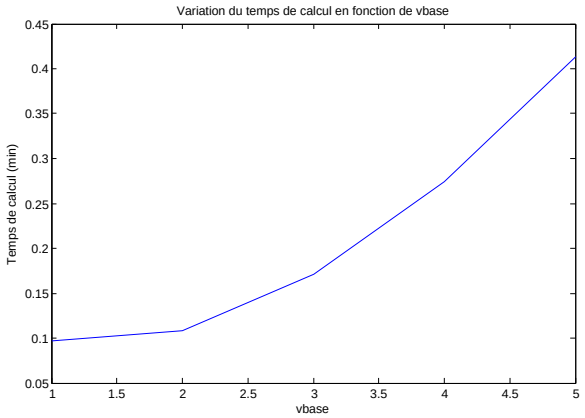


FIGURE 4.25 – Variation du temps de calcul en fonction de *vbase*

Interprétations :

Ensembles :

- De même, que pour la première base, on constate la diminution des taux d'erreur quand on ajoute des visages de clients dans la base d'apprentissage.
- Le taux TFA est très petit, parfois nul dans la base de test.

Temps de calcul :

- Le temps de calcul augmente toujours en ajoutant des visages dans la base.

2) PCA_RNA

De même, nous essayerons toujours d'améliorer l'algorithme PCA en introduisant le réseau de neurones multicouches pour faire la reconnaissance de visages.

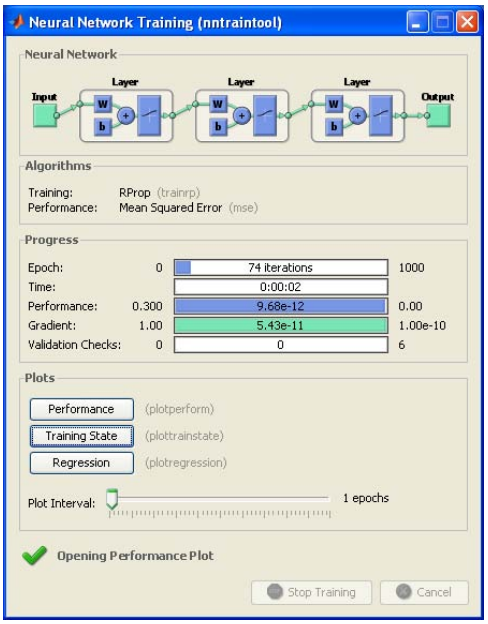
Plusieurs expériences ont été faites jusqu'à l'obtention du réseau convenable.

Après les tests, nous avons choisi de garder 2 réseaux ; le premier comporte 40 sorties telle que la sortie i soit mise à un et les autres mises à 0 pour indiquer l'indentification en faveur de la personne **i**.

Le deuxième réseau comporte 6 sorties, pour coder les nombres de 1 à 40 en binaire.

1. ***Le premier réseau***

   Ce réseau contient 5 entrées puisqu'on utilise 5 visages propres, deux couches cachées qui contiennent respectivement 128 et 200 neurones, la sortie contient 40 neurones et la fonction d'activation utilisée dans les différentes couches est : **logsig**. L'algorithme d'apprentissage utilisé est **trainrp** et l'apprentissage a pris 74 itérations :



   La figure 4.26 donne la variation de l'erreur quadratique en fonction du nombre d'itérations :

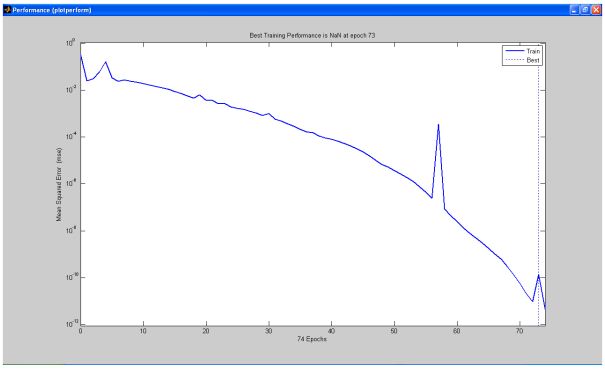


FIGURE 4.26 – Variation de l'erreur quadratique en fonction du nombre d'itérations

Evaluation et comparaison des performances du réseau

Les résultats obtenus sont représentés sur les tableaux 4.17, 4.18, 4.19 et 4.20, en les comparant avec les résultats de la distance euclidienne.

– **L'ensemble d'évaluation** : (Clients : 40×4/ Imposteurs : 20×5)

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	11.9231	11.9231
TFR(%)	11.9231	11.9231
(TFA+TFR) (%)	23.8462	23.8462

TABLE 4.17 – Comparaison des taux de l'ensemble d'évaluation entre PCA et PCA_RNA

– **L'ensemble de test** :(Clients :40×3/ Imposteurs : 20×6)

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	6.6667	7.0833
TFR(%)	8.7500	8.75001
(TFA+TFR) (%)	15.4167	15.8333

TABLE 4.18 – Comparaison des taux de l'ensemble de test entre PCA et PCA_RNA

– **Le temps de calcul** :

Distance euclidienne	RNA
0.0971 min	0.0451 min

TABLE 4.19 – Comparaison du temps de calcul entre PCA et PCA_RNA

– **3^{e} ensemble** : (Clients de l'ensemble d'évaluation)

Méthode / Nbre d'images par client	Distance euclidienne (Taux (%))	RNA (Taux(%))
1	17.5000	15
2	15	17.5000
3	14.1667	17.5000
4	15	18.1250

TABLE 4.20 – Comparaison du temps de calcul entre PCA et PCA_RNA

Interprétations :

– **L'ensemble d'évaluation** :
Le réseau réalisé a donné les mêmes performances que celles de la distance euclidienne.

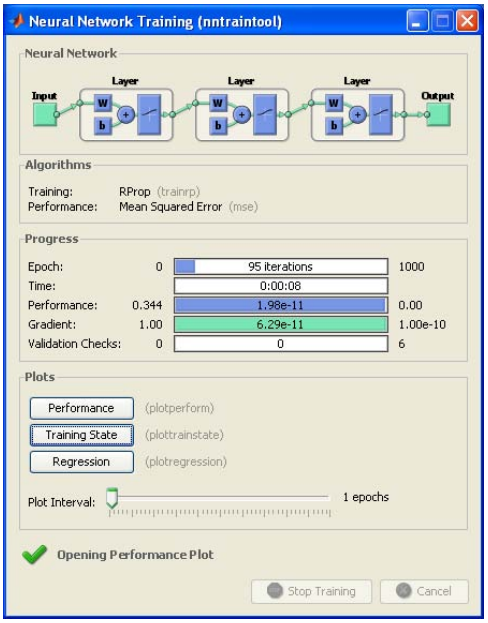
– **L'ensemble de test** :
On a eu une différence de 0.4165% (une personne) seulement pour le TFA.

– **Le temps de calcul** :
Le temps de calcul du réseau de neurones est toujours meilleur que celui de la distance euclidienne.

– **Dernier ensemble** :
On remarque que c'est la seule fois où le réseau de neurones donne un taux d'erreur meilleur que celui de la distance euclidienne et ceci quand on prend une seule image par client.
Pour les autres cas, les taux de la distance euclidienne sont toujours meilleurs que ceux du réseau, quoique la différence soit petite.

2. ***Le deuxième réseau***

La seule différence entre ce réseau et le précédent est le nombre de sorties. Sinon, l'architecture (nombre de couches, neurones et fonctions de transfert) est la même. Ce réseau contient 6 sorties et l'apprentissage a pris 95 itérations :



La figure 4.27 donne la variation de l'erreur quadratique en fonction du nombre d'itérations :

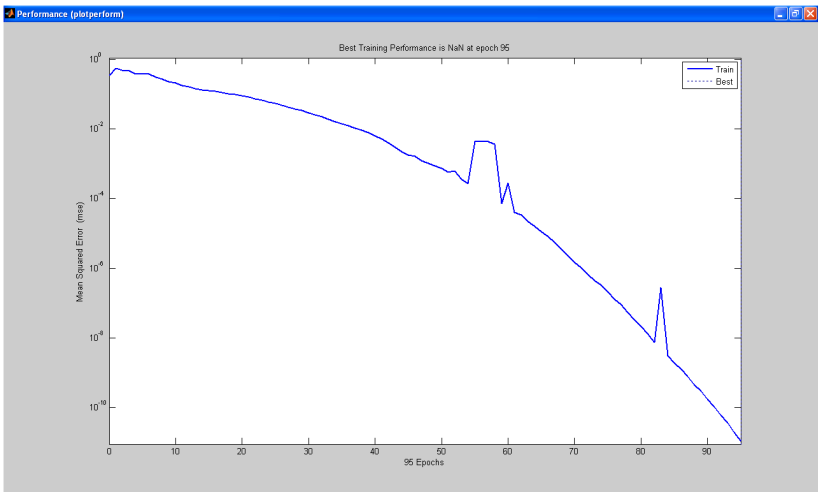


FIGURE 4.27 – Variation de l'erreur quadratique en fonction du nombre d'itérations

Evaluation et comparaison des performances du réseau

Les résultats obtenus sont représentés sur les tableaux 4.21, 4.22 et 4.23 , en les comparant avec les résultats de la distance euclidienne.

– **L'ensemble d'évaluation** :

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	11.9231	18.8462
TFR(%)	11.9231	11.9231
(TFA+TFR) (%)	23.8462	30.7693

TABLE 4.21 – Comparaison des taux de l'ensemble d'évaluation entre PCA et PCA_RNA

– **L'ensemble de test** :

Méthode / Taux	Distance euclidienne	RNA
TFA(%)	6.6667	10.0000
TFR(%)	8.7500	8.7500
(TFA+TFR) (%)	15.4167	18.7500

TABLE 4.22 – Comparaison des taux de l'ensemble de test entre PCA et PCA_RNA

– **Le temps de calcul** :

Distance euclidienne	RNA
0.0971 min	0.0429 min

TABLE 4.23 – Comparaison du temps de calcul entre PCA et PCA_RNA

Interprétations :

– **L'ensemble d'évaluation** :
La distance euclidienne a donné des résultats meilleurs cette fois-ci. La différence entre les TFA est de 6.9231% (18 personnes).

– **L'ensemble de test** :
On a eu une différence de 3.3333% (8 personnes) pour le TFA, toujours en faveur de la distance euclidienne.

– **Le temps de calcul** :
Le temps de calcul du réseau de neurones est toujours meilleur que celui de la distance euclidienne.

4.4.3 Troisième base (40 personnes) avec des images compressées (25×23)

Pour compresser les images, nous avons utilisé une fonction créée par nos collègues qui travaillent aussi sur un sujet de reconnaissance de visages.

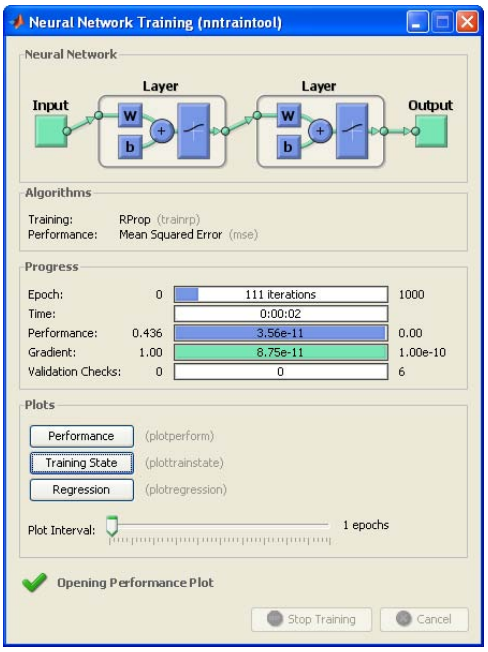
L'algorithme qu'utilise cette fonction pour compresser les images et réduire leur résolution de (200×180) à (25×23) est le **WPD** (Wavelet Packet Decomposition) qui veut dire **Décomposition en Paquets d'ondelettes**.

Nous avons fait des tests sur l'ensemble d'évaluation seulement pour régler les paramètres pour la partie VHDL.

La distance euclidienne et le réseau de neurones multicouche ont donné des taux identiques et qui sont :

$$\begin{cases} \text{TFA} = 11.5385\% \\ \text{TFR} = 11.9231\% \end{cases}$$

Le réseau réalisé contient 5 entrées, une seule couche cachée constituée de 56 neurones, la couche de sortie est constituée de 40 neurones et la fonction d'activation utilisée dans les deux couches (cachée, sortie) est **logsig**. L'algorithme d'apprentissage utilisé est le **trainrp** et l'apprentissage a pris 111 itérations :



La figure 4.28 donne la variation de l'erreur quadratique en fonction du nombre d'itérations :

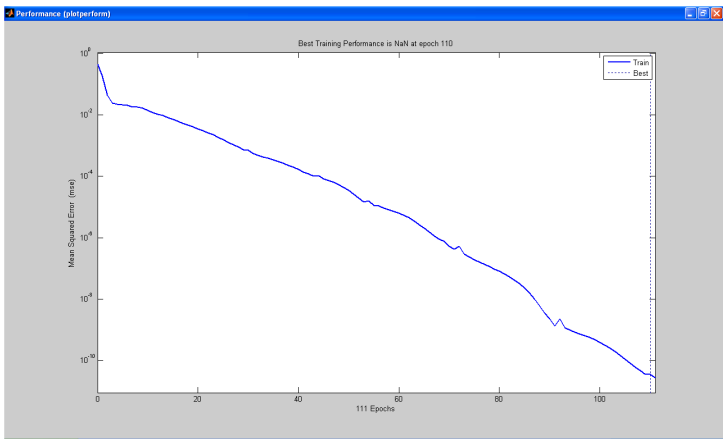


FIGURE 4.28 – Variation de l'erreur quadratique en fonction du nombre d'itérations

4.5 Conclusion

Matlab est un logiciel de calcul très puissant. L'ensemble des boîtes à outils qu'il propose (tel, Neural Network Toolbox, Image Processing Toolbox, ...) contribue largement à sa puissance .

La simulation sur Matlab nous a permis de valider le fonctionnement de notre système. Nous avons fait varier plusieurs paramètres lors des tests afin d'obtenir les meilleurs résultats possibles.

Ainsi, la simulation sur Matlab permet de gagner beaucoup de temps et de tester le bon fonctionnement des systèmes avant l'investissement final dans la fabrication.

Chapitre 5

Implémentation sur cible matérielle (FPGA) du système de reconnaissance réalisé

5.1 Introduction

Dans cette partie, nous allons présenter l'implémentation que nous avons faite pour notre système de reconnaissance de visages.

D'abord, Il faut noter les choix que nous avons fait pour cette implémentation. Concernant le langage de programmation, nous avons opté pour le langage de description matérielle VHDL qui est l'un des langages les plus utilisés dans ce domaine. Le circuit d'implémentation est un

circuit FPGA **XC4VFX12** de la famille **Virtex-4** de Xilinx. Pour l'outil de programmation, nous avons choisi l'environnement de développement de Xilinx qui est ISE (version 10.1).

La présentation de ces choix fera l'objet de la section **2**.

Après présentation des outils utilisés, et dans la section **3**, nous allons commencer par expliquer l'architecture de notre système, en décrivant ces principales composantes et le rôle de chaque partie. On verra aussi les principales étapes de programmation ainsi que les principaux problèmes et contraintes rencontrés durant la réalisation du programme.

Dans la section **4**, on présentera l'interface de simulation que nous avons créé à l'aide de Matlab, afin de faire l'adaptation des données d'entrées qui sont les visages des personnes à l'entrée du système et aussi pour permettre de voir les résultats de cette simulation sur cette interface.

La section **5**, fera l'objet de la présentation des résultats et de l'évaluation de l'implémentation.

5.2 Présentation des outils, circuits et langages utilisés

5.2.1 Les circuits FPGA

Un circuit logique programmable, ou réseau logique programmable, est un circuit intégré logique qui peut être reprogrammé après sa fabrication.

Il est composé de nombreuses cellules logiques élémentaires librement assemblables. Ce type de composant électronique est communément désigné par les sigles anglais :

- **FPGA** (field-programmable gate array, réseau de portes programmables *in situ*),
- **PLD** (programmable logic device, circuit logique programmable),
- **EPLD** (erasable programmable logic device, circuit logique programmable et effaçable),
- **CPLD** (complex programmable logic device, circuit logique programmable complexe),
- **PAL** (programmable array logic, réseau logique programmable),
- **PLA** (programmable logic array, réseau logique programmable),
- Etc.

Bien que fondamentalement synonymes, ces termes ne sont généralement pas interchangeables dans le vocabulaire commercial des fabricants : FPGA désigne plutôt des composants à technologie RAM, EPLD des composants à technologie FLASH, PAL des composants à technologie fusible.

Les **réseaux logiques programmables** sont des circuits composés de nombreuses cellules logiques élémentaires librement assemblables.

Celles-ci sont connectées de manière définitive ou réversible par programmation, afin de réaliser la ou les fonctions numériques voulues. L'intérêt est qu'une même puce peut être utilisée dans de nombreux systèmes électroniques différents.

Certains modèles peuvent aussi comporter : de la mémoire d'usage général, des blocs « DSP » câblés, des boucles à verrouillage de phase pour la génération d'horloge.

Les FPGA

La plupart des grands FPGA modernes sont basés sur des cellules SRAM aussi bien pour le routage du circuit que pour les blocs logiques à interconnecter.

Un bloc logique est de manière générale constitué d'une table de correspondance (LUT ou *Look-Up-Table*) et d'une bascule (*Flip-Flop* en anglais). La LUT sert à implémenter des équations logiques ayant généralement 4 à 6 entrées et une sortie. Elle peut toutefois être

considérée comme une petite mémoire, un multiplexeur ou un registre à décalage. Le registre permet de mémoriser un état (machine séquentielle) ou de synchroniser un signal (*pipeline*).

FIGURE 5.1 – Exemple d'un circuit FPGA de type Virtex-5 de la famille Xilinx

Les blocs logiques, présents en grand nombre sur la puce (de quelques milliers à quelques millions en 2007) sont connectés entre eux par une matrice de routage configurable. Ceci permet la reconfiguration à volonté du composant, mais occupe une place importante sur le silicium et justifie le coût élevé des composants FPGA. La topologie est dite « Manhattan », en référence aux rues à angle droit de ce quartier de New York.

Les densités actuelles ne permettent plus un routage manuel, c'est donc un outil de placement-routage automatique qui fait correspondre le schéma logique voulu par le concepteur et les ressources matérielles de la puce. Comme les temps de propagation dépendent de la longueur des liaisons entre cellules logiques, et que les algorithmes d'optimisation des placeurs-routeurs ne sont pas déterministes, les performances (fréquence max.) obtenues dans un FPGA sont variables d'un design à l'autre.

L'utilisation des ressources est par contre très bonne, et des taux d'occupation des blocs logiques supérieures à 90 % sont possibles.

Comme la configuration (routage et LUTs) est faite par des points mémoire volatiles, il est nécessaire de sauvegarder le design du FPGA dans une mémoire non volatile externe, généralement une mémoire Flash série, compatible « JTAG ». Certains fabricants se distinguent toutefois par l'utilisation de cellules EEPROM pour la configuration, éliminant le recours à une mémoire externe, ou par une configuration par anti-fusibles (la programmation par une tension élevée fait « claquer » un diélectrique, créant un contact). Cette dernière technologie n'est toutefois pas reconfigurable.

Quelques fonctionnalités particulières disponibles sur certains composants :
- blocs de mémoire supplémentaires (hors des LUT), souvent double-port, parfois avec mécanisme de FIFO,
- multiplieurs câblés (coûteux à implémenter en LUT), voire blocs multiplieur-accumulateur pour traitements DSP,
- cœur de microprocesseur enfoui (dit hard core),
- blocs PLL pour synthétiser ou resynchroniser les horloges,
- reconfiguration partielle, même en cours de fonctionnement,
- chiffrement des données de configuration,
- sérialiseurs/désérialiseurs dans les entrées-sorties, permettant des liaisons série haut-débit,
- impédance contrôlée numériquement dans les entrées-sorties, évitant de nombreux composants passifs sur la carte.

Afin de pouvoir finaliser un FPGA, il est nécessaire d'utiliser un langage de description matériel ou bien un outil de saisie graphique. Après compilation de cette description, on obtient un fichier de configuration pour le FPGA choisi.

Les applications

Les FPGA sont utilisés dans diverses applications nécessitant de l'électronique numérique (télécommunications, aéronautique, transports...). Ils sont également utilisés pour le prototypage d'ASIC.

Les FPGA sont généralement plus lents, plus chers à l'unité et consomment d'avantage d'énergie que leur équivalent en ASIC (*Application Specific Integrated Circuit*). Cependant, ils ont plusieurs avantages :

- délai de mise sur le marché plus court, car ce sont des composants standards,
- temps de développement plus court, car on réutilise des fonctions de base et la reconfigurabilité autorise une validation préalable moins stricte,
- coût inférieur pour de petites séries (moins de 10 000 unités). Avec l'évolution technologique, cette quantité tend à augmenter : en effet, le prix d'une puce est proportionnel à sa surface, qui diminue avec la finesse de gravure, tandis que les coûts initiaux pour fabriquer un ASIC (conception, tests, masques de gravure) sont en forte augmentation.

Il est parfois possible de transformer directement un FPGA en une version ASIC plus rapide, moins chère et consommant moins (car les matrices de routage sont remplacées par une couche de métallisation fixe).

Plusieurs FPGA modernes possèdent la possibilité d'être reconfigurés (on parle de configuration lorsqu'il s'agit de programmation du matériel) partiellement à la volée. Ceci permet d'obtenir des systèmes reconfigurables - par exemple une unité centrale dont les instructions changent dynamiquement en fonction des besoins.

Les FPGA modernes sont assez vastes et contiennent suffisamment de mémoire pour être configurés pour héberger un cœur de processeur ou un DSP, afin d'exécuter un logiciel. On parle dans ce cas de processeur softcore, par opposition aux microprocesseurs hard-core enfouis dans le silicium.

Aujourd'hui, les fabricants de FPGA intègrent même un ou plusieurs cœurs de processeur « hard-core » sur un même composant afin de conserver les ressources logiques configurables du composant.

Ceci n'exclut pas l'utilisation de processeur softcore possédant de nombreux avantages. On tend donc vers des « Systems On Chip », comme pour le microcontrôleur il y a quelques décennies, avec en plus de la logique configurable selon l'utilisateur. La mémoire des tous derniers FPGA est encore insuffisante pour exécuter des logiciels embarqués un peu complexes et on doit avoir recours à des mémoires externes (ROM, RAM). Cependant, la loi de Moore n'est pas encore à bout de souffle et celles-ci devraient être intégrées dans quelques années et suffiront à une grande partie des applications embarquées.

5.2.2 Le langage VHDL

VHDL (VHSIC Hardware Description Langage) est un langage de description de matériel, c'est-à-dire un langage utilisé pour décrire un système numérique matériel, comme, par exemple, un flip-flop (bascule D) ou un microprocesseur. Il peut modéliser un système par n'importe quelle vue, structurelle ou comportementale, à tous les niveaux de description.

De plus il peut servir non seulement à simuler un système mais aussi à le synthétiser, c'est-à-dire être transformé par des logiciels adaptés (synthétiseurs) en une série de portes logiques prêtes à être gravées sur du silicium.

VHDL est l'un des trois grands langages de description de matériel utilisés majoritairement

dans l'industrie, avec Verilog et SystemC. Chaque langage a ses propres avantages et inconvénients, ainsi que ses spécificités. Pour plus de détails, vous pouvez consulter la référence [26].

Le langage standard IEEE VHDL a été développé par le Groupe d'Analyse et de Standardisation VHDL (VASG, pour « VHDL Analysis and Standardization Group »). Larry Saunders est le coordinateur de VASG. La société CLSI (CAD Langage Systems Inc.), représentée par le Docteur Moe Shahdad et M. Erich Marschner a préparé une série d'analyses et de recommandations dont a été tirée en Février 1986 la version 7.2 de VHDL, point de départ du futur standard. La collaboration de CLSI au projet était financée par un contrat passé avec l'Air Force Wright Aeronautical Laboratories, représentée par le Docteur John Hines. Le standard définitif a été adopté vers le milieu de l'année 1987. [27]

Afin de répondre aux différents problèmes de l'électronique, la norme VHDL a dû évoluer. L'IEEE Design Automation Standards Committee (DASC) a créé la norme IEEE 1076.1 (1999), ou VHDL-AMS (VHDL-Analog & Mixed Systems). Cette nouvelle norme est une extension de la norme IEEE 1076-1987 déjà existante. Elle permet la description et la simulation de circuits analogiques, numériques, et mixtes (analogique et numérique). Pour cela elle utilise en complément des instructions séquentielles et concurrentes un nouveau type d'instructions, dites « simultanées », et qui ont valeur d'équations. En pratique, de plus en plus de simulateurs implémentent cette extension. Par contre, les outils de synthèse analogique associés n'en sont encore qu'à leurs balbutiements. [28]

La syntaxe du VHDL est tirée du langage Ada, dont les mots clefs ont été adaptés à la conception matérielle. L'une des particularités du VHDL provient du fait qu'il est possible d'exprimer facilement le parallélisme présent à l'intérieur d'un circuit.

En VHDL, tout composant (dans le sens logiciel) est décrit sous deux aspects :
- L'interface avec le monde extérieur, décrite dans une section dénommée **entity**.
- L'implémentation elle-même, décrite dans une section dénommée **architecture**.

C'est donc l'**ARCHITECTURE** qui contient la description de la fonction matérielle désirée :
- soit sous forme de description structurelle précise de l'architecture matérielle (les portes logiques à utiliser et leurs interconnexions),
- soit sous forme de comportement attendu, c'est-à-dire orienté fonctionnel. [28]

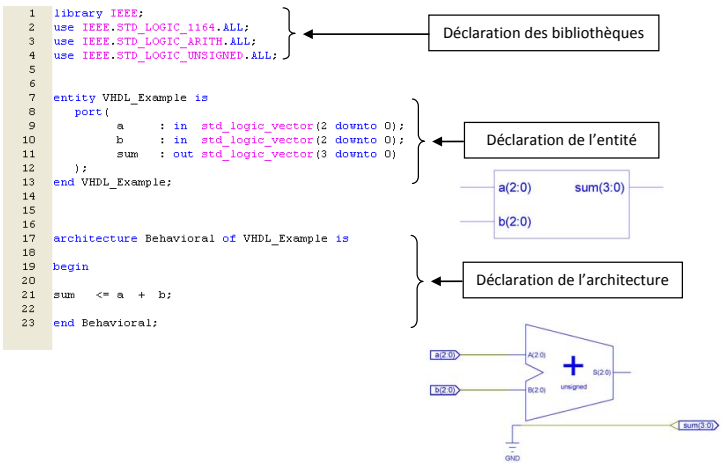


FIGURE 5.2 – structure générale d'un programme VHDL

Simulation et synthèse

Le but d'un langage de description matérielle tel que le VHDL est de faciliter le développement d'un circuit numérique en fournissant une méthode rigoureuse de description du fonctionnement et de l'architecture du circuit désirée. L'idée est de ne pas avoir à réaliser (fondre) un composant réel, en utilisant à la place des outils de développement permettant de vérifier le fonctionnement attendu. Ce langage permet en effet d'utiliser des simulateurs, dont le rôle est de tester le fonctionnement décrit par le concepteur.

L'étape suivante consiste à synthétiser cette description matérielle pour obtenir un composant réalisant les fonctions désirées, à l'aide d'éléments logiques concrets (portes logiques, bascules ou registres). Ceux-ci seront implémentés, selon la technologie utilisée, soit directement en transistors (dans le cas d'un ASIC), ou en se basant sur les éléments programmables des FPGA. Après la synthèse viennent les phases de :

- placement : on choisit l'emplacement physique des différents éléments ;
- routage : on détermine les connexions entre éléments.

Ces deux opérations doivent prendre en compte les ressources disponibles sur l'ASIC (surface) ou dans le FPGA (unités programmables).

Le VHDL ayant une double fonction (simulation et synthèse), une partie seulement du VHDL est synthétisable, l'autre existant uniquement pour faciliter la simulation (écriture de modèles comportementaux et de test benches). Selon le support matériel et le logiciel de synthèse utilisés, cette partie pourra être plus ou moins étendue. De manière à obtenir du VHDL synthétisable et portable, il est donc nécessaire de se limiter à des constructions simples, dont la transcription en portes et bascules est simple à réaliser. La norme 1076.6 a été initiée pour tenter de définir un sous-ensemble de VHDL « de synthèse ». [28]

5.2.3 L'outil ISE

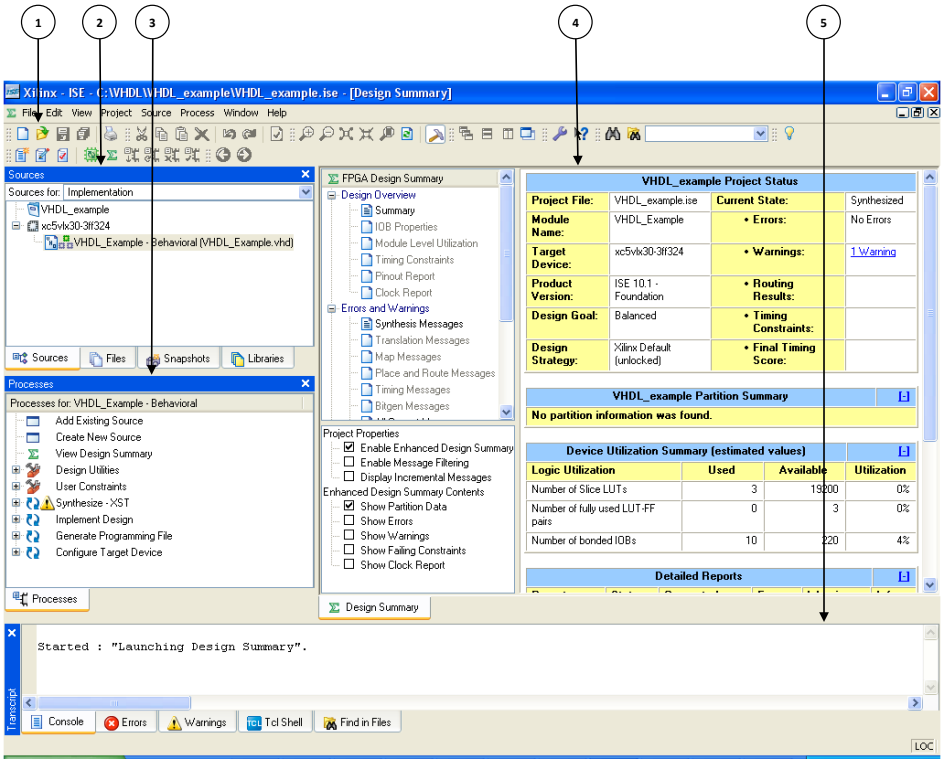
ISE ou Integrated Software Environment (ISE^{TM}) est le logiciel de conception de circuits de Xilinx qui nous permet de suivre les étapes de réalisation de notre système de l'écriture de la description matérielle, en passant par la synthèse jusqu'à l'implémentation sur la cible matérielle. Le ISE Project Navigator va nous guider à développer et à suivre l'évolution de notre système à travers les différentes étapes du flot de conception (ISE design flow).

Il est à noter que pour notre projet, nous avons utilisé la version 10.1 d'ISE.

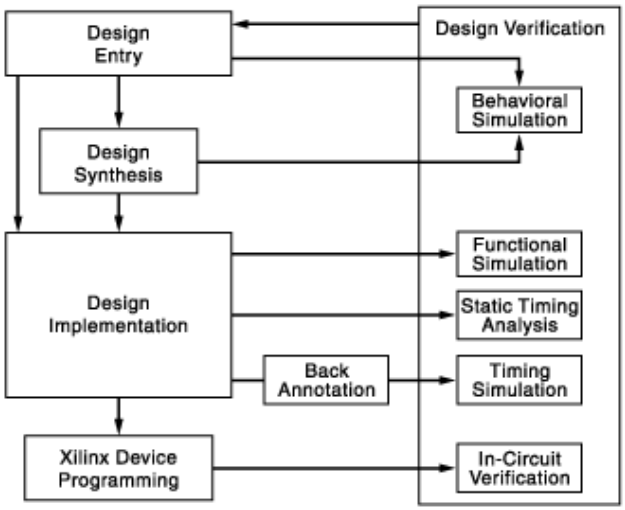
Présentation de l'interface graphique d'ISE

L'interface graphique d'ISE est constituée principalement des éléments suivants :

1. Barre d'outils :
   Donne un accès rapide aux commandes fréquemment utilisées.
2. Fenêtre des sources :
   Cette fenêtre permet d'afficher les différents fichiers de notre projet. Elle permet aussi (Sources for) de choisir l'approche de la conception, soit d'un point de vue implémentation ou simulation.
3. Fenêtre des Processus :
   Cette fenêtre permet le lancement des actions (processes) sur le fichier choisi dans la fenêtre des sources. A noter que les différentes actions sont classées dans un ordre hiérarchique suivant le déroulement typique d'un flot de conception (section suivante).
4. Workspace :
   Quand nous ouvrons un fichier source ou exécutons certaines actions, c'est là où l'affichage correspondant va se faire.
5. Fenêtre Transcript :
   Cette fenêtre affiche les messages (erreurs, warnings, ...) résultants des actions que nous faisons. Dépendamment des fichiers que nous choisissons et des outils utilisés, on voit apparaitre des onglets additionnels dans cette fenêtre. [29]



Etapes d'ISE design flow



Source : [29]

- **Entrée du design (Design Entry)** :
  C'est la première des étapes dans le flow de conception. Durant cette étape, nous créons nos fichiers sources basés sur les objectifs de notre circuit. Nous pouvons créer un fichier de haut niveau utilisant un langage de description matérielle (Hardware Description Language -HDL-) comme VHDL, Verilog ou ABEL ou sinon en utilisant un schématique. Comme nous pouvons utiliser plusieurs formats pour les fichiers sources de bas niveau dans notre design.
- **Synthèse (Design Synthesis)** :
  Après l'entrée du design nous passons à la synthèse. Durant cette étape, les fichiers VHDL et Verilog ou autres deviennent des fichiers netlist qui sont acceptés comme des entrées pour l'étape de l'implémentation.
  Notons aussi que si nous travaillons avec un fichier EDIF ou NGC/NGO synthétisé nous pouvons sauter les deux premières étapes et aller directement à l'implémentation.
- **Implémentation (Design Implementation)** :
  Après la synthèse, nous exécutons design implementation, qui convertit le design (fichier) logique en un format de fichier physique qui pourra être chargé dans le circuit cible. A partir du navigateur de projets (Project Navigator) nous pouvons exécuter le processus d'implémentation en une seule étape ou en plusieurs. Les processus de l'implémentation varient suivant que nous ciblons un FPGA (Field Programmable Gate Array) ou un CPLD (Complex Programmable Logic Device).
- **Vérification (Design Verification)** :
  Nous pouvons vérifier le fonctionnement de notre design ou circuit en différents points dans le flot de conception. Nous pouvons utiliser un logiciel de simulation (ModelSim, Xilinx ISE Simulator,...) pour vérifier le fonctionnement et le timing de notre design ou une portion de celui-ci.
  Le simulateur interprète le code VHDL ou Verilog en fonctionnement du circuit et donne des résultats logiques pour le HDL décrit afin de déterminer l'opération correcte du circuit. La simulation nous permet de créer et vérifier les fonctions complexes dans un temps relativement court. Nous pouvons aussi faire une vérification directe sur le circuit après l'avoir programmé.

– **Configuration du circuit (Device Configuration)** :
Après avoir généré un fichier de programmation, nous configurons notre circuit. Durant la configuration, nous générons des fichiers de configuration et chargeons les fichiers de programmation dans le circuit à partir de notre ordinateur.

5.3 Structure du système de reconnaissance réalisé

Dans cette partie nous allons voir la structure de notre système. D'abord, nous verrons l'implémentation de l'algorithme PCA et les éléments constituants de l'architecture. Puis, nous allons présenter l'implémentation de notre réseau de neurones multicouches. Nous présenterons aussi les quelques modifications à faire pour obtenir un programme synthétisable et quelques spécificités de programmation.

5.3.1 Structures pour la simulation

Les structures présentées ici pour le système réalisé sont plus destinées à la simulation qu'à l'implémentation. Bien qu'elles ne soient pas assez différentes des structures destinées à l'implémentation d'un point de vue externe, les structures destinées à la simulation doivent être adaptées pour être synthétisables.

1. **Structure de l'algorithme PCA**

   Avant d'entamer la description de la structure, il faut mentionner quelques spécifications sur l'ensemble des données utilisées.
   D'abord, pour cette structure, nous avons essayé de prendre une base de données constituée de 150 personnes avec le choix de 10 visages propres. Mais, en entamant la programmation et vu les temps que prendrait la simulation et la difficulté de vérification des résultats, nous avons décidé de diminuer cette base jusqu'à 40 personnes et de prendre 5 visages propres au lieu de 10 ce qui nous a aidé à valider l'implémentation.
   Il est à noter aussi que la résolution des images est de 200×180 (36000 pixels). Bien que ce soit une basse résolution, elle nous a créé des problèmes lors de la synthèse ce qui nous a amené plus tard à la diminuer pour l'obtention d'un modèle synthétisable. Cet effet et d'autres seront développés plus en détail dans la section dénommée **Structures pour l'implémentation**.

   Pour l'implémentation de l'algorithme PCA, nous avons adopté une structure composée de deux parties, qui sont respectivement le bloc Mat_mul et le bloc Dist_eucl, dont voici la structure générale (figure 5.3) :

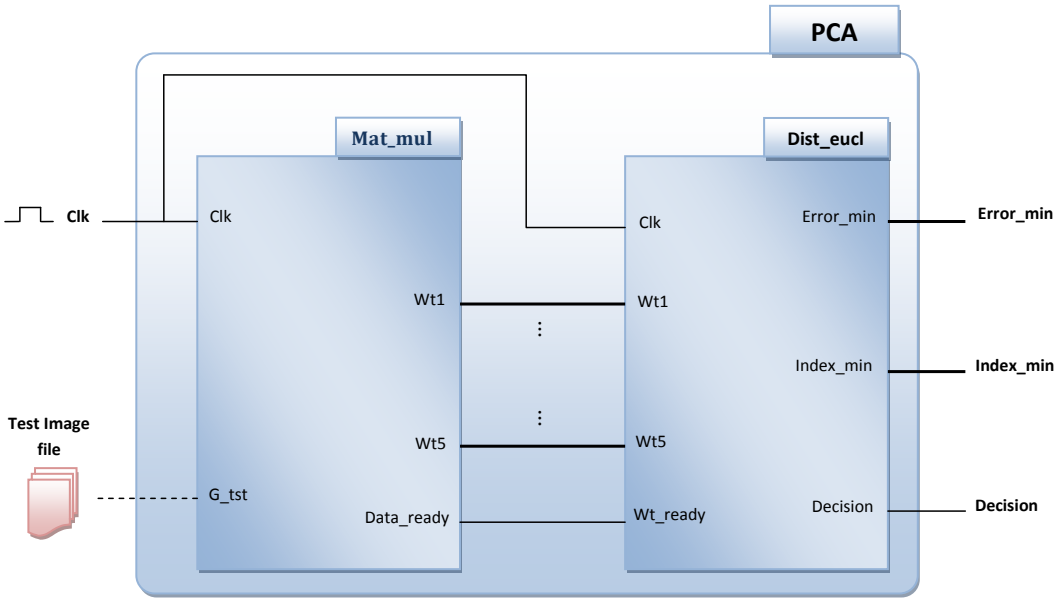


FIGURE 5.3 – Structure générale de l'implémentation de l'algorithme PCA

Note :

Les fils les plus épais représentent des bus de données (données sur plusieurs bits) tandis que ceux moins épais représentent des données sur un bit.

Explications

La structure de l'algorithme PCA est composée d'un bloc qui a deux entrées (Clk, Test image file) et trois sorties (Error_min, Index_min et Decision).

Les entrées :

- **Clk** : C'est l'horloge externe qui va synchroniser tous les éléments de la structure. En général, elle est issue de l'oscillateur qui se trouve sur la carte qui contient le circuit FPGA.
- **Test image file** : C'est l'entrée mise en pointillés. Elle représente le fichier qui contient l'image de la personne à tester. En fait, ce n'est pas un fil ou un bus qui va être raccordé à la structure mais plutôt, un fichier de données créé à l'aide du logiciel de calcul Matlab et inclus dans la structure à l'aide d'une fonction de lecture créée dans le programme VHDL.

Ce choix a été fait pour plusieurs raisons, parmi lesquelles, on peut citer :

- Adapter et faciliter l'entrée au système.
- Alléger la structure et ne pas ajouter un nouveau bloc de conversion de l'image issue éventuellement d'une caméra.
- Eviter plusieurs contraintes de programmation VHDL tel que l'interfaçage entre l'élément d'acquisition externe et la structure.
- Valider rapidement le fonctionnement du système et éliminer d'éventuelles causes d'erreurs.

Il est à noter, que cette entrée peut être remplacée plus tard par un bloc qui fera l'adaptation entre l'élément d'acquisition (caméra, appareil photo,...) et l'entrée de la structure.

La création de ce fichier passe essentiellement par les étapes suivantes :
- Acquisition de l'image de la personne et la mettre dans une matrice ;
- Conversion des éléments de la matrice en niveaux de gris ;
- Concaténation de la matrice en un seul vecteur ;
- Codage des éléments de ce vecteur en binaire suivant leur gamme de variation.

Ainsi, notre fichier contiendra l'image de la personne arrangée comme un vecteur d'éléments binaires. Cette étape de conversion et codage est très importante car c'est elle qui va en quelque sorte remplacer la bonne ou mauvaise qualité de l'acquisition.

Les sorties :
- **Error_min** : Cette sortie représente la distance euclidienne minimale trouvée après avoir calculé toutes les distances euclidiennes entre les poids représentatifs de l'image de test et ceux de la base.
- **Index_min** : Donne l'indice de la personne qui a donné le minimum de distance euclidienne. Donc, la personne la plus proche de la personne de test.
- **Decision** : C'est la sortie qui donne la décision sur la personne, c.à.d. si la personne est un client ou un imposteur en se basant sur un seuil.

Mat_mul :
Maintenant, nous allons nous intéresser au premier bloc qui constitue la structure PCA. Ce bloc est représenté à la figure 5.4.

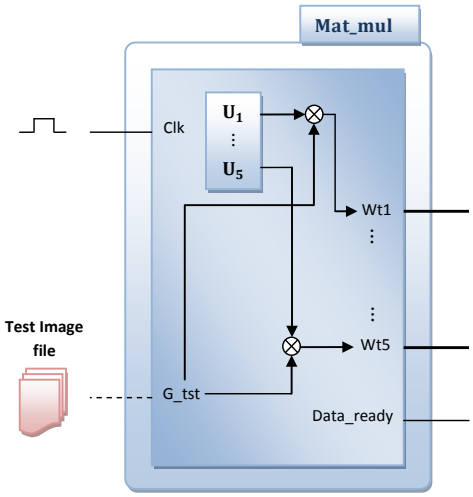


FIGURE 5.4 – Structure du bloc Mat_mul

Les entrées :
- **Clk** et **Test image file** décrites précédemment.

Les sorties :
- **Wt1** à **Wt5** : Sont les poids représentatifs de l'image de test.
- **Data_ready** : Est un signal qui indique la fin des opérations et la présence des résultats finaux sur les sorties.

Le bloc Mat_mul va faire la multiplication matricielle $Wt = U^t \cdot G\_tst$, avec G_tst un vecteur représentant l'image de test et qui est de dimension (36000×1) et U^t la transposée de la matrice U de dimension (36000×5).

Cette matrice U est la matrice qui contient les 5 visages propres issus de l'exécution de l'algorithme PCA dans Matlab.

La multiplication matricielle va nous permettre d'extraire les poids représentatifs de l'image de test et qui seront utilisés ensuite pour calculer les distances euclidiennes dans le bloc suivant.

Ainsi, le résultat sera le vecteur Wt (poids de l'image de test) de dimension (5×1).

Pour faire cette multiplication matricielle, nous l'avons décortiqué en plusieurs multiplications matricielles plus simples afin de l'adapter à la programmation, et ceci, en remarquant que :

$$Wt = \begin{pmatrix} Wt1 = U_1 \cdot G\_tst \\ Wt2 = U_2 \cdot G\_tst \\ Wt3 = U_3 \cdot G\_tst \\ Wt4 = U_4 \cdot G\_tst \\ Wt5 = U_5 \cdot G\_tst \end{pmatrix} \tag{5.1}$$

Avec

$$U = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{pmatrix} \tag{5.2}$$

Ces multiplications sont synchronisées par le signal d'horloge de telle sorte que l'on fera pour chaque sortie une multiplication élémentaire à chaque changement du front d'horloge.

Dist_eucl

Nous allons, maintenant, voir la structure du bloc Dist_eucl (figure 5.5).

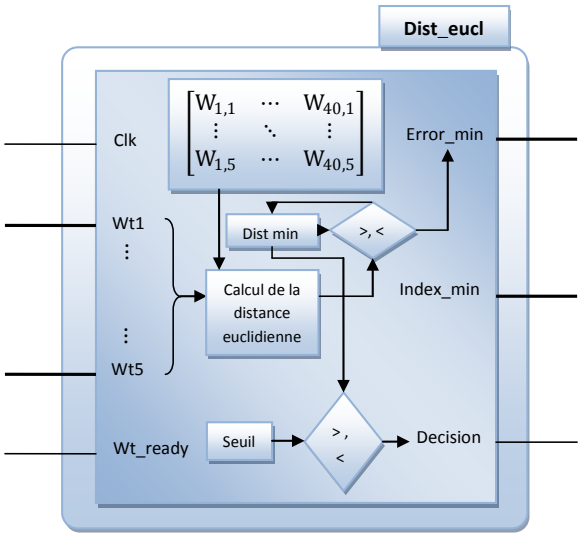


FIGURE 5.5 – Structure du bloc Dist_eucl

Les entrées :

- **Clk** : Entrée pour la synchronisation du système.
- **Wt1** à **Wt5** : Sont les poids représentatifs de l'image de test obtenus du bloc Mat_mul.
- **Wt_ready** : Lié à la sortie Data_ready du bloc Mat_mul, il indique la fin des calculs de ce dernier.

Les sorties :

- **Error_min**, **Index_min** et **Decision** : Sont les sorties du bloc général PCA.

Ce bloc permet le calcul de la distance euclidienne entre le vecteur poids Wt représentatif de l'image de test et des vecteurs de poids W stockés au préalable dans des roms et qui représentent les poids représentatifs de chaque visage de la base de données.

A chaque changement du front de l'horloge, ce bloc calcul la distance entre Wt et un des poids de la base et vérifie si cette distance est plus petite que celle qui la précède. Si c'est le cas, le bloc garde cette distance et l'indice du poids de la base correspondant. Et ainsi de suite jusqu'à l'obtention après 40 itérations (nombre de visages dans la base de données) à la sortie **Error_min**, la distance euclidienne minimale et à la sortie **Index_min**, l'indice du visage qui donne cette distance minimale.

Pour la sortie **Decision**, elle est par défaut mise à 1 indiquant que la personne correspondante est un client. La détection des imposteurs se fait grâce à un seuil qui permettra de positionner la sortie **Decision** à 0.

Il faut noter que le seuil est défini expérimentalement après un certain nombre d'essais.

2. **Structure de la combinaison PCA_RNA**

Dans cette partie, nous exposerons la structure du système réalisé. Cette structure se base sur l'exploitation des éléments constituant la structure PCA vue dans la section précédente et sur un bloc RNA qui représentera notre réseau de neurones artificiels multicouche.

Voici la structure générale de notre système de reconnaissance :

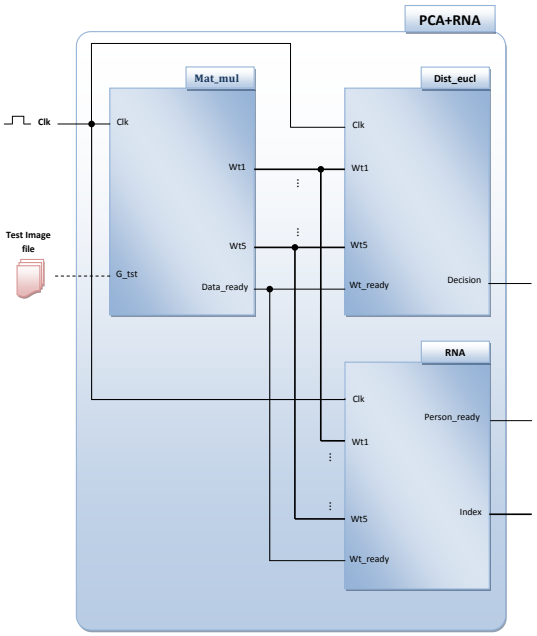


FIGURE 5.6 – Structure générale du système de reconnaissance réalisé

Les entrées :

– **Clk** et **Test image file** décrites précédemment.

Les sorties :

– **Decision** : C'est la seule sortie gardée dans le bloc Dist_eucl. Nous avons choisi de la laisser à cause de la structure de notre réseau de neurones multicouche qui ne peut pas fournir une décision sur la catégorie (client ou imposteur) du visage à reconnaître.
– **Person_ready** : Cette sortie indique que le résultat de la reconnaissance est prêt.
– **Index** : Donne l'indice de la personne de la base de données la plus proche de la personne à reconnaître.

Dans la section suivante, on va s'intéresser au bloc RNA.

RNA

C'est le bloc de notre réseau de neurones multicouche. Notre réseau est constitué de 3 couches (une couche d'entrée, une couche de sortie et une couche cachée). La première couche contient 128 neurones, la deuxième contient 200 neurones et la dernière contient 40 neurones. Chaque neurone est suivi d'une fonction de transfert de type **Log-sigmoid**.

Voici le schéma général de notre réseau :

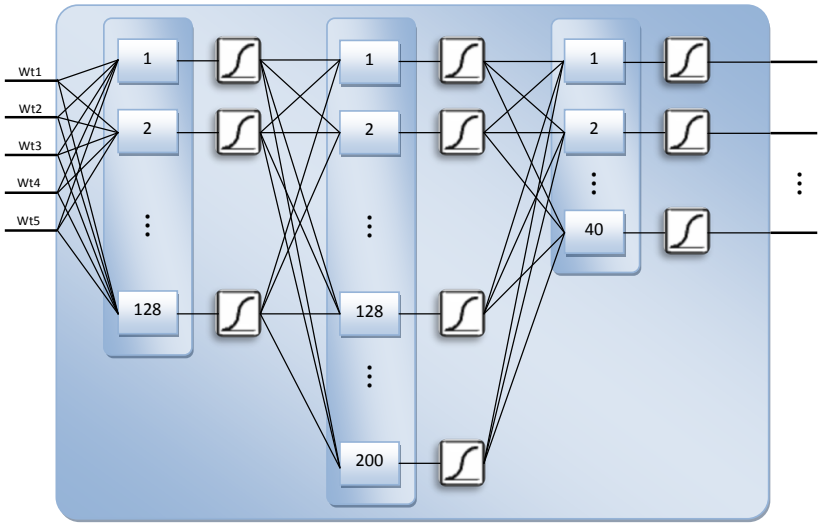


FIGURE 5.7 – Schéma général de notre réseau de neurones multicouches

Pour l'implémentation, nous avons sauvegardé les poids et les biais (qui sont issus du programme Matlab) de notre réseau dans des roms afin de les utiliser dans le calcul des valeurs des sorties.

Les données à sauvegarder sont :

IB : Les 128 biais de la couche d'entrée

IW : Les poids de la couche d'entrée

LB2 : Les 200 biais de la deuxième couche

LW2 : Les poids de la deuxième couche

LB3 : Les 40 biais de la dernière couche

LW3 : Les poids de la dernière couche.

Dans notre programme, le début des opérations d'une couche sera conditionné par la fin des opérations de celle qui la précède.

Pour les fonctions de transferts, nous les avons implémentées en tant que LUT (Look Up Table) pour éviter le passage par la fonction exponentielle et la division.

5.3.2 Structures pour l'implémentation

Comme il a était évoqué dans la section **Structures pour la simulation**, les structures destinées à l'implémentation ne sont qu'une adaptation des structures destinées à la simulation.

Après avoir essayé de faire la synthèse (l'outil de synthèse est XST : Xilinx Synthesis Tool incorporé dans ISE) de notre système dont la structure a été explicitée dans la section précédente nous n'avions pas eu de résultats et nous étions amené à suspecter plusieurs paramètres dont le premier a été la résolution de nos images, ce qui nous a conduit à la diminuer de 200×180 à 25×23. Cette modification engendre la modification de la structure de notre réseau de neurones (la structure du réseau et le changement de la résolution sont mentionné dans la section **Troisième base (chap4)**).

Ce nouveau réseau contient une couche cachée composée de 56 neurones et une couche de sortie contenant 40 neurones.

Nous avons également, enlevé les fonctions d'écriture (écriture de l'indice et de la décision) car elles ne sont pas synthétisables. Comme nous avons du aussi changer des boucles for, utilisées dans la programmation du réseau de neurones, par des boucles construites avec le test conditionnel if car le synthétiseur se bloquait et ne donnait pas de résultats.

La LUT de la fonction d'activation a été changée aussi où nous avons diminué sa taille de 1964 éléments à 102. Notant que malgré cette grande différence en nombre, les résultats n'ont pas changé beaucoup.

Pour vérifier si les composants de notre système sont tous synthétisables, nous avons lancé la synthétisation (outil : XST) de chaque composant seul et ceci en choisissant le circuit FPGA Virtex-4 de Xilinx XC4VFX12. Voici les résultats obtenus pour chaque composant.

Mat_mul :

```
=========================================================================
*                            Final Report                               *
=========================================================================
Final Results
RTL Top Level Output File Name     : mat_mul.ngr
Top Level Output File Name         : mat_mul
Output Format                      : NGC
Optimization Goal                  : Area
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 187

Cell Usage :
# BELS                             : 5531
```

```
#      GND                          : 1
#      INV                          : 5
#      LUT1                         : 33
#      LUT2                         : 39
#      LUT3                         : 1018
#      LUT4                         : 2929
#      MUXCY                        : 43
#      MUXF5                        : 878
#      MUXF6                        : 321
#      MUXF7                        : 157
#      MUXF8                        : 75
#      VCC                          : 1
#      XORCY                        : 31
# FlipFlops/Latches                 : 418
#      LD                           : 418
# Clock Buffers                     : 1
#      BUFG                         : 1
# IO Buffers                        : 186
#      OBUF                         : 186
# DSPs                              : 5
#      DSP48                        : 5
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 4vfx12sf363-12

 Number of Slices:                     2185  out of   5472    39%
 Number of Slice Flip Flops:            232  out of  10944     2%
 Number of 4 input LUTs:               4024  out of  10944    36%
 Number of IOs:                         187
 Number of bonded IOBs:                 186  out of    240    77%
    IOB Flip Flops:                     186
 Number of GCLKs:                         1  out of     32     3%
 Number of DSP48s:                        5  out of     32    15%
```

Dist_eucl :

```
=========================================================================
*                            Final Report                               *
=========================================================================
Final Results
RTL Top Level Output File Name     : dist_eucl.ngr
Top Level Output File Name         : dist_eucl
Output Format                      : NGC
Optimization Goal                  : Area
Keep Hierarchy                     : NO
```

```
Design Statistics
# IOs                              : 274

Cell Usage :
# BELS                             : 9531
#      BUF                         : 1
#      GND                         : 1
#      INV                         : 130
#      LUT1                        : 149
#      LUT2                        : 1269
#      LUT3                        : 493
#      LUT4                        : 1323
#      MULT_AND                    : 667
#      MUXCY                       : 2873
#      MUXF5                       : 35
#      MUXF6                       : 6
#      VCC                         : 1
#      XORCY                       : 2583
# FlipFlops/Latches                : 578
#      LDE                         : 578
# Clock Buffers                    : 1
#      BUFG                        : 1
# IO Buffers                       : 273
#      IBUF                        : 186
#      OBUF                        : 87
# DSPs                             : 24
#      DSP48                       : 24
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 4vfx12sf363-12

 Number of Slices:                     1750  out of   5472    31%
 Number of Slice Flip Flops:            491  out of  10944     4%
 Number of 4 input LUTs:               3364  out of  10944    30%
 Number of IOs:                         274
 Number of bonded IOBs:                 273  out of    240   113% (*)
    IOB Flip Flops:                      87
 Number of GCLKs:                        1  out of     32     3%
 Number of DSP48s:                      24  out of     32    75%

WARNING:Xst:1336 -  (*) More than 100% of Device resources are used
```

RNA :

```
=========================================================================
*                            Final Report                               *
=========================================================================
Final Results
RTL Top Level Output File Name     : RNA.ngr
Top Level Output File Name         : RNA
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : NO

Design Statistics
# IOs                              : 194

Cell Usage :
# BELS                             : 65049
#      BUF                         : 29
#      GND                         : 1
#      INV                         : 114
#      LUT1                        : 188
#      LUT2                        : 732
#      LUT2_D                      : 72
#      LUT2_L                      : 5
#      LUT3                        : 16302
#      LUT3_D                      : 2505
#      LUT3_L                      : 91
#      LUT4                        : 22340
#      LUT4_D                      : 273
#      LUT4_L                      : 1197
#      MUXCY                       : 9237
#      MUXF5                       : 6773
#      MUXF6                       : 2708
#      MUXF7                       : 1190
#      MUXF8                       : 476
#      VCC                         : 1
#      XORCY                       : 815
# FlipFlops/Latches                : 8885
#      LD                          : 1051
#      LDE                         : 7834
# Clock Buffers                    : 1
#      BUFG                        : 1
# IO Buffers                       : 193
#      IBUF                        : 186
#      OBUF                        : 7
# DSPs                             : 32
#      DSP48                       : 32
=========================================================================
```

```
Device utilization summary:
---------------------------

Selected Device : 4vfx12sf363-12

 Number of Slices:                    24399  out of   5472   445% (*)
 Number of Slice Flip Flops:           8884  out of  10944    81%
 Number of 4 input LUTs:              43819  out of  10944   400% (*)
 Number of IOs:                         194
 Number of bonded IOBs:                 193  out of    240    80%
    IOB Flip Flops:                       1
 Number of GCLKs:                         1  out of     32     3%
 Number of DSP48s:                       32  out of     32   100%

WARNING:Xst:1336 -  (*) More than 100% of Device resources are used
```

Remarques : Bien que nous ayons réussi à synthétiser les composants, on voit que la carte Virtex-4 choisie ne supporte pas notre réseau de neurones qui prend beaucoup de ressources.

On voit aussi dans le rapport du composant Dist_eucl qu'il prend plus d'IOBs (Input Output Blocks) que ce que nous fournit le circuit FPGA. En fait, ce n'est pas un problème, car ce composant sera par la suite incorporé comme un des éléments du système et on n'aura à faire qu'aux IOBs globaux de ce dernier qui seront plutôt moins nombreux.

5.3.3 Quelques spécificités de programmation

Dans cette section, nous allons voir quelques spécificités de notre programmation VHDL.

D'abord, il est primordial de faire un bon choix du type de données utilisées dans la programmation.

VHDL présente au programmeur une multitude de types prédéfinis, parmi lesquels on peut citer : integer, bit, bit_vector, std_logic, std_logic_vector, real, time, boolean, etc. Cependant, on trouve des types qui sont destinées seulement aux simulations et ne sont pas synthétisables sur des circuits physiques. Donc, le programmeur doit veiller à ce que les types de données soient bien adaptés à ses besoins.

Vu la nécessité de types synthétisables et bien adaptés à nos calculs, nous avons opté pour la bibliothèque (library) **ieee_proposed** qui contient la définition des types **sfixed** et **ufixed** dans le paquetage (package) **fixed_pkg**.

Ces deux types représentent des nombres codés en virgule fixe respectivement signés (signed fixed point) et non signés (unsigned fixed point).

Voici, un exemple de la définition d'un signal qui représente un nombre signé codé en virgule fixe avec 4 bits pour la partie entière et 2 bits pour la partie fractionnaire.

Signal **Nbre_sign** : sfixed (3 downto -2) ;

Les bits de la partie entière sont les bits de 3 à 0, avec comme bit de signe le bit3. Et les bits -1, -2 sont ceux de la partie fractionnaire.

Le paquetage **fixed_pkg** définit aussi plusieurs opérations, et même des fonctions qui aident à redimensionner les données et à faire des transformations vers d'autres types de données.

Un avantage très important de ce paquetage est que les fonctions qui y sont définies pour l'addition, soustraction, multiplication et division prennent en compte les bits de carry et les bits de signe.

Néanmoins, il faut bien veiller à spécifier la taille des résultats des opérations d'après les règles définies dans le paquetage. Ces règles peuvent être trouvées dans la référence [30]. Voici, un exemple d'une opération d'addition :

Signal **Nbre_sign1** : sfixed (3 downto -2) ;
Signal **Nbre_sign2** : sfixed (3 downto -2) ;
Signal Rés_add :sfixed(4 downto -2) ; –Rés_add=Nbre_sign1+Nbre_sign2

On remarque ici que la partie entière du résultat contient 5 bits tandis que la partie fractionnaire en contient 2 (d'après les règles du paquetage, Rés_add_MSB=max(Nbre_sign1_MSB, Nbre_sign1_MSB)+1 et Rés_add_LSB=min(Nbre_sign1_LSB, Nbre_sign1_LSB)).

Un autre point à mentionner est la façon de programmer les roms. On définit d'abord un nouveau type qui sera un tableau de valeurs. Puis, on déclare la rom comme une constante de ce type et on lui affecte les valeurs. Voici un exemple illustratif d'une rom qui contient 3 éléments :

type rom_type is array of (0 to 2) of sfixed (0 downto -4) ;
constant rom : rom_type :=(¨ 10111¨ ,¨ 01010¨ ,¨ 11011¨) ;

L'accès aux éléments se fera par l'indice de celui-ci. Exemple : rom(2) donne accès à « 01010 ».

L'avantage de cette programmation est qu'elle facilite la manipulation des éléments des roms.

Nous avons aussi tiré parti des fonctions que VHDL permet de créer. Ainsi, nous avons créé des fonctions de lecture et d'écriture de données, que nous avons utilisées pour lire le fichier Test image file qui ne peut pas être mis dans une rom et aussi pour écrire les résultats (indice et décision) dans des fichiers pour pouvoir les affichées dans notre interface d'aide à la simulation crée avec Matlab.

Une remarque à faire sur le paquetage utilisé est qu'il nous a créé autant de problèmes qu'il nous a fourni d'aide.

En fait, les problèmes rencontrés viennent essentiellement des règles strictes à suivre dans la définition de la taille des signaux pour chaque opération. A l'exemple de l'opération MAC (Multiplication ACcumulation) qui est essentielle pour faire la multiplication matricielle et les calculs des sorties des couches du réseau de neurones, nous ne pouvions pas par exemple écrire directement résultat=résultat+quelque chose. Car les tailles des signaux sont fixes et le paquetage indique pour l'addition que la partie entière du résultat doit être augmentée d'un bit par rapport au maximum des MSB des deux opérandes de l'opération. Et puisque un des opérandes de cette opération est lui-même le résultat, on tombe dans une ambigüité.

Alors, nous avons pallier cette contrainte par une petite astuce qui est de mettre résultat=résultat(MSB-1)+quelque chose. L'astuce est donc de définir un bit supplémentaire dans la taille du résultat. Ce bit ne sera jamais atteint lors des opérations et ainsi pour l'opérande résultat, on ne garde que les autres bits. L'inconvénient, est qu'on doit connaître avec précision la gamme de variation de nos résultats afin de prendre une décision sur les tailles et sur le bit supplémentaire.

5.4 Présentation de l'interface d'aide à la simulation

Cette interface a été créée pour la simulation du fonctionnement des programmes VHDL réalisés. Donc, elle nous facilite la gestion des entrées et des sorties et permet de simuler le déroulement de la procédure d'identification.

5.4.1 Vue globale

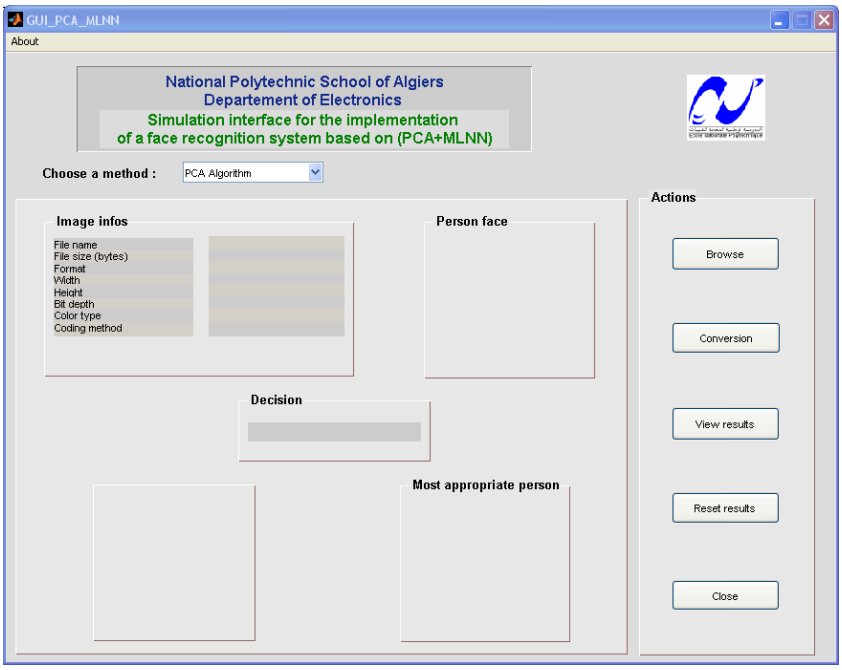


FIGURE 5.8 – Vue globale de l'interface réalisée

Notre interface est composée des éléments suivants :

1. **Menu About** : Ce menu donne quelques informations sur l'interface

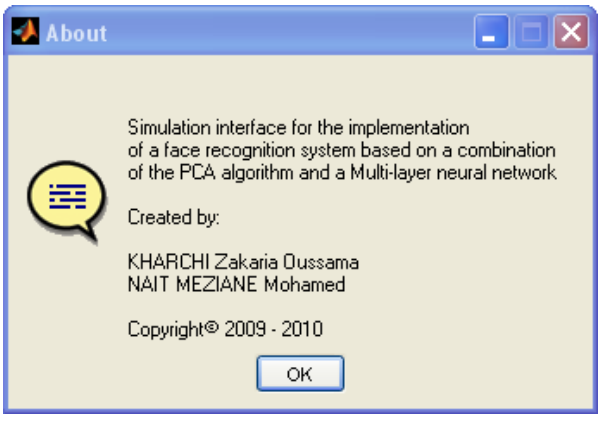


FIGURE 5.9 – Menu About

2. **Une proposition de choix d'une méthode** : C'est un menu déroulant qui propose le choix entre l'algorithme PCA et le système réalisé avec la combinaison du réseau de neurones multicouche.

FIGURE 5.10 – Proposition de choix d'une méthode

3. **Boutons d'action** : Ces boutons permettent d'exécuter les actions durant l'identification.

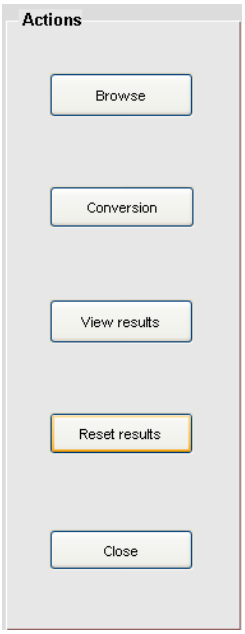


FIGURE 5.11 – Boutons d'action

4. **Espace d'affichage des résultats** : C'est l'espace réservé aux affichages des résultats, messages et informations durant l'identification.

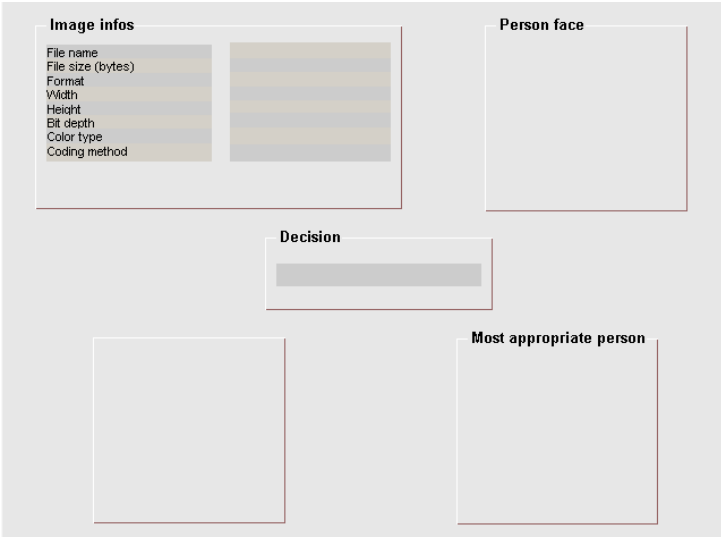


FIGURE 5.12 – Espace d'affichage des résultats

5.4.2 Déroulement d'une identification

– ***Choose a method*** : Choix d'une méthode pour l'identification.

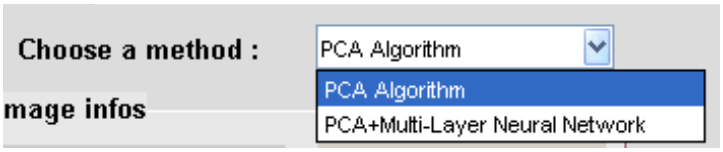


FIGURE 5.13 – Choix d'une méthode pour 'identification

– ***Browse*** : Choix d'une personne à identifier.

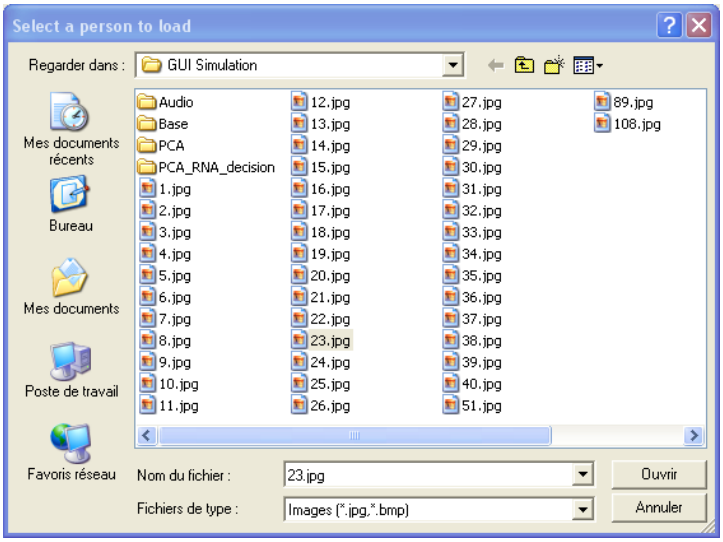


FIGURE 5.14 – Choix d'une personne à identifier

Après le choix d'une personne, voici le résultat obtenu :

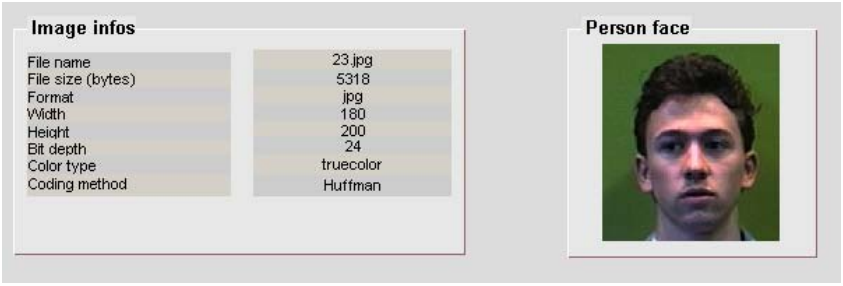


FIGURE 5.15 – Affichage de l'image de test avec l'interface graphique

Nous avons mis une petite tablette qui donne quelques informations sur l'image à identifier.

– *Conversion* : La conversion est lancée par le bouton **Conversion**. Cette conversion génère le fichier Test image file adapté à l'entrée de notre programme VHDL. Nous avons ajouté une barre qui indique la progression de la conversion.

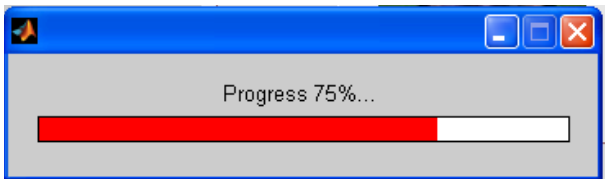


FIGURE 5.16 – Progression d'une conversion

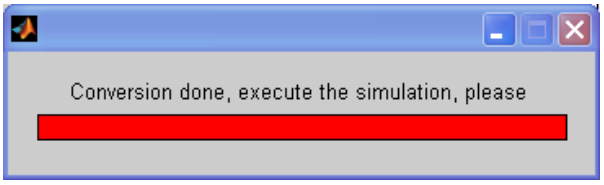


FIGURE 5.17 – Fin de la conversion

A la fin de la conversion, l'interface graphique lance automatiquement le logiciel ISE, en demandant d'exécuter la simulation.

FIGURE 5.18 – Lancement automatique du logiciel ISE

– *Exécution de la simulation* : Après l'exécution de la simulation, voici le résultat que donne le simulateur intégré à ISE.

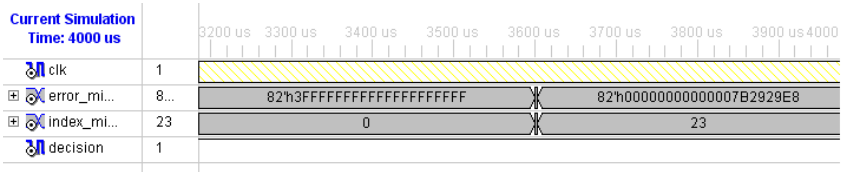


FIGURE 5.19 – Résultat simulé avec ISE

– *View results* : En revenant maintenant à notre interface graphique et en appuyant sur le bouton View results, on voit apparaître les résultats de notre identification avec acceptation ou refus de la personne et aussi en indiquant la personne de la base qui est la plus proche de cette personne.

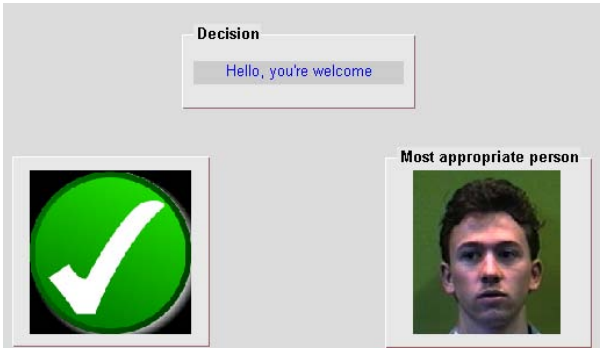


FIGURE 5.20 – Identification faite avec l'interface graphique

– *Reset* : Bouton pour la réinitialisation de l'identification.
– *Close* : Bouton qui permet de fermer l'interface graphique.

5.4.3 Améliorations de l'interface

Nous avons ajouté quelques améliorations dans l'interface pour la rendre plus performante. En voici quelques unes :

Les messages d'erreur :

FIGURE 5.21 – Message d'erreur (lancement de la conversion avant faire rentrer l'image de test)

Ce message est généré si on essaie de faire la conversion sans choisir une personne pour l'identification.

FIGURE 5.22 – Message d'erreur (Demande des résultats avant de faire la conversion)

Ce message est généré au cas où on lance la vision des résultats avant la conversion.

Les sons :

Nous avons ajouté quelques sons audio qui viennent accompagner les messages et le déroulement de la simulation.

5.5 Présentation des résultats

Bonne acceptation :

Résultat ISE :

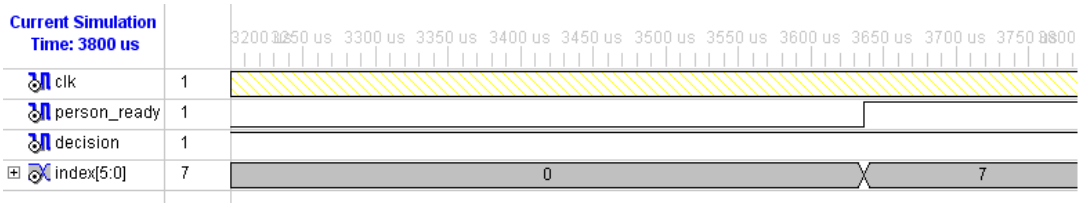


FIGURE 5.23 – Résultat ISE (Bonne acceptation)

Résultat interface Matlab :

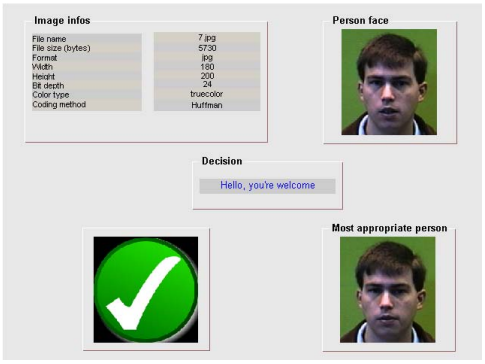


FIGURE 5.24 – Résultat interface Matlab (Bonne acceptation)

Fausse acceptation :

1. ***Imposteur accepté*** :

   Résultat ISE :

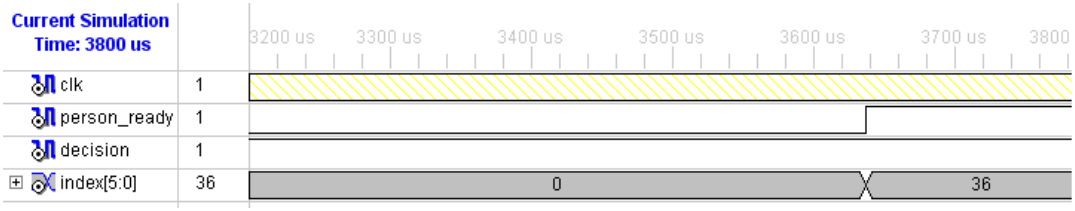


FIGURE 5.25 – Résultat ISE (Imposteur accepté)

Résultat interface Matlab :

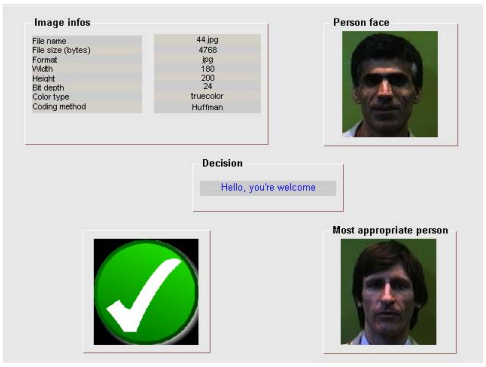


FIGURE 5.26 – Résultat interface Matlab (Imposteur accepté)

2. ***Client mal identifié*** :

Résultat ISE :

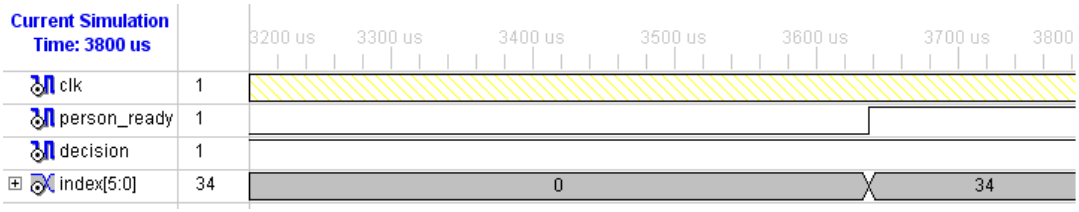


FIGURE 5.27 – Résultat ISE (Client mal identifié)

Résultat interface Matlab :

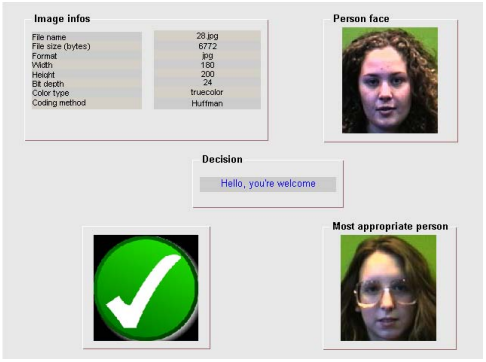


FIGURE 5.28 – Résultat interface Matlab (Client mal identifié)

Bon rejet :

Résultat ISE :

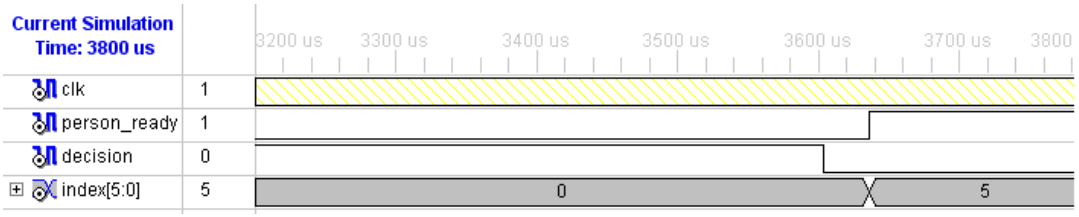


FIGURE 5.29 – Résultat ISE (Bon rejet)

Résultat interface Matlab :

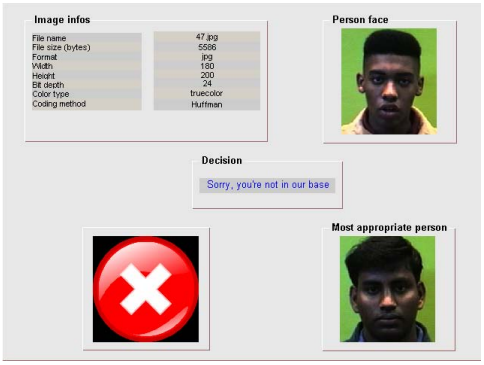


FIGURE 5.30 – Résultat interface Matlab (Bon rejet)

Faux rejet :

Résultat ISE :

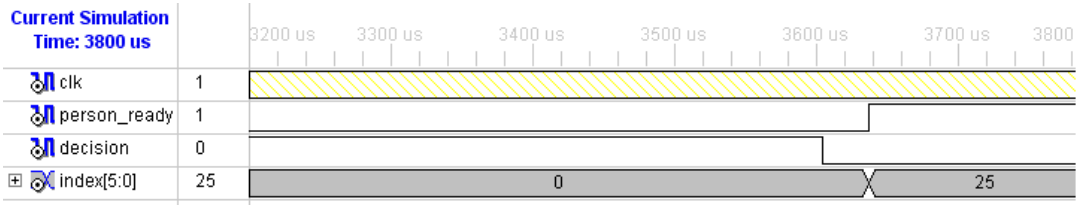


FIGURE 5.31 – Résultat ISE (Faux rejet)

Résultat interface Matlab :

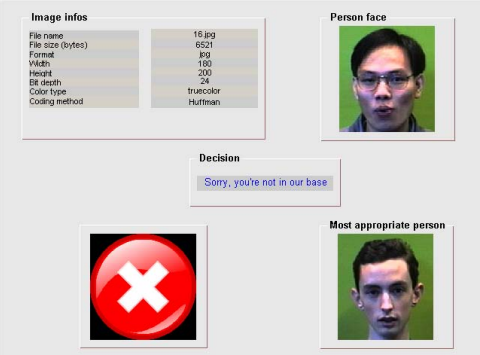


FIGURE 5.32 – Résultat interface Matlab (Faux rejet)

5.6 Conclusion

Les résultats obtenus précédemment montrent les différents cas possibles que peut donner notre système lors de l'identification d'un visage. Et nous pouvons ainsi conclure qu'il fonctionne normalement.

D'après les rapports de synthèse générés par l'outil ISE, nous remarquons que notre système est bien synthétisable. Malheureusement, notre cible matérielle n'a pas pu le contenir vu ses ressources limitées par rapport aux exigences des composants du système et essentiellement ceux du réseau de neurones qui demande beaucoup d'espace mémoire et d'opérations à faire.

Il est à noter aussi que d'après les temps de simulation, nous pouvons dire que notre système est un système temps réel.

Conclusion et perspectives

L'algorithme PCA avec la distance euclidienne pour la décision donne un bon taux de reconnaissance, mais ce taux reste toujours insuffisant pour les systèmes stricts. Ce taux peut être amélioré en utilisant plus d'images pour chaque client dans la base de données. Mais ce choix influencera beaucoup sur le temps de calcul. Une autre solution pour améliorer ce taux sera d'augmenter le nombre de visages propres gardés, mais arrivant à une certaine limite, l'ajout de visages propres ne portera plus aucune information supplémentaire et retardera par contre les calculs.

L'algorithme PCA avec un réseau de neurones multicouche nous a permis de faire l'identification, mais il ne nous a pas donné une amélioration dans les taux par rapport à ceux de l'algorithme PCA seul (Décision par utilisation de la distance euclidienne) hormis peut être le temps de calcul qui a diminué. Les taux de reconnaissance du système réalisé sont, malgré cela, proches des taux de reconnaissance évalués avec l'algorithme PCA seul, où nous avons trouvé un taux d'erreur de 21,5% sur une base de données contenant 150 personnes et ceci avec le test de 760 visages.

Il est à noter aussi que le système réalisé fait l'identification par le réseau de neurones, mais il ne peut donner en sortie que l'indice d'un des clients de sa base, ce qui nous a amené à utiliser la distance euclidienne pour détecter les imposteurs.

Concernant l'implémentation, nous pouvons donner les constatations suivantes :

- Le réseau de neurones nécessite beaucoup d'espace mémoire pour le stockage des biais, poids et la LUT de la fonction d'activation.
- Le choix du type et du codage des signaux influe beaucoup sur la précision des résultats.
- Le seuil de décision doit être ajusté pour tenir compte de la divergence entre la simulation et l'implémentation causée par le choix du codage.

Toutes ces constatations, que nous avons faites, nous amènent à conclure que le réseau de neurones reste malgré ses insuffisances une bonne alternative pour l'identification.

Enfin, pour optimiser davantage notre système, nous proposons comme perspectives de :

- Essayer d'améliorer l'algorithme PCA par la combinaison avec un réseau de neurones autre que le multicouche.
- Essayer d'optimiser la programmation matérielle du réseau de neurones afin qu'il nécessite moins de ressources.
- Ajouter un bloc VHDL qui fera l'acquisition des visages directement d'une caméra.

Bibliographie

[1] M. KOCHER. Murat KUNT, G. GRANLUND. *Traitement numérique des images, volume 2.* presse polytechnique et universitaires normandes, 1993.

[2] BOUABIDA Naima. Segmentation des tissus cérébraux sur des images par résonance magnétique. Projet de fin d'études, Electronique, ENP, Alger, 2007.

[3] FRADJ Mohammed Karim, LOUKAL Abdelkader. Segmentation par étiquetage des artères coronaires avec implémentation sur DSP C6000. Projet de fin d'études, Electronique, ENP, Alger, 2007.

[4] KADDOUR Chakib, AISSA BRAHIM Salim. Généralités sur le traitement d'images. http://www.kaddour.com/chap1/chap1.htm.

[5] DOUMANDJI Samah, BOULFANI Yasmine. Implémentation sur DSP TMS320C5000 de filtres optimaux appliqués aux images et introduction de réseaux neuronaux. Projet de fin d'études, Electronique, ENP, Alger, 2004.

[6] S.PHILIPP J.COCQUEREZ. *Analyse d'image : filtrage et segmentation.* éditions MASSON, 1995.

[7] Didier GUILLERM. Introduction à la biométrie. http://www.biometrie-online.net/presentation.

[8] Bowman, Eric. Everything you need to know about biometrics. http://www.ibia.org/EverythingAboutBiometrics.PDF, January 2000. Identix Corporation.

[9] Bernadette DORIZZI, Philippe LAMADELAINE, Claudine GUERRIER et Jean LEROUX LES JARDINS. *La biométrie, Techniques et usages.* Techniques de l'ingénieur. H 5 530.

[10] Florent PERRONNIN et Jean-Luc DUGELAY. *Introduction à la Biométrie, Authentification des Individus par Traitement Audio-Vidéo.* Institut Eurécom, Multimedia Communications Department, 2002.

[11] Didier GUILLERM. Les modalités biométriques. http://www.biometrie-online.net/technologies/les-modalites.

[12] http://www.linternaute.com/science/biologie/dossiers/06/0607-biometrie/retine.jpg.

[13] REDA JOURANI. Reconnaissance de visages. Diplôme des etudes supérieures approfondies, Informatique et Télécommunications, La Faculté des Sciences de Rabat, 2006.

[14] Djallel Chefrour Mohamed Tayeb Laskri. *Who_Is : système d'identification des visages humains.* A R I M A - Volume 1 - 2002, pages 39 à 61.

[15] W. Zhao, R. Chellappa, P. J. Phillips et A. Rosenfeld. *Face Recognition : A Literature Survey.* ACM Computing Surveys. December 2003. Vol. 35, No. 4, pages 399 à 458.

[16] Xiaoyin xu. "image based face recognition using global features". http://www.vlsi.uwindsor.ca/RCIM3/Seminar.html. Research Centre for Integrated Microsystems Electrical and Computer Engineering, University of Windsor, Supervisor : Dr. Ahmadi. Seminar, May 13, 2005.

[17] http://www.biometricgroup.com.

[18] Didier GUILLERM. Les modalités biométriques. http://www.biometrie-online.net/le-marche.

[19] Nicolas MORIZET. Reconnaissance Biométrique par Fusion Multimodale du Visage et de l'Iris. http://pastel.paristech.org/5811/, Ecole Doctorale d'Informatique,Télécommunications et Electronique de Paris, March 2009. Thèse de doctorat, Spécialité : Signal et Images.

[20] Lindsay I Smith. A tutorial on Principal Components Analysis. http://www.cs.otago.ac.nz/cosc453/student_tutorials/.

[21] http://mathcentral.uregina.ca/RR/database/RR.09.95/weston2.html.

[22] Adrien Vergé. Réseaux de neurones artificiels. http://www.k-netweb.net/projects/tipe/.

[23] Dr Libor Spacek. Computer Vision Science Research Projects, Description of the Collection of Facial Images. http://cswww.essex.ac.uk/mv/allfaces/.

[24] Juergen Luettin et Gilbert Maître. Evaluation Protocol for the M2VTS Database (XM2VTSDB). IDIAP COMMUNICATION, July 1998.

[25] Howard Demuth, Mark Beale et Martin Hagan. *Neural Network Toolbox ™ 6, User's Guide, Matlab.* sec. 5-50.

[26] Douglas J. Smith. VHDL & Verilog Compared & Contrasted Plus Modeled Example Written in VHDL, Verilog and C. http://www.angelfire.com/in/rajesh52/verilogvhdl.html.

[27] Alexis Polti, COMELEC 2005. Cours en ligne VHDL, Introduction, historique. http://comelec.enst.fr/hdl/vhdl_intro.html.

[28] Wikipedia, L'encyclopédie libre. http://fr.wikipedia.org/wiki/VHDL#cite_note-2.

[29] Logiciel Xilinx ISE 10.1, Xilinx ISE help.

[30] David Bishop. Fixed point package user's guide. www.vhdl.org/fphdl/Fixed_ug.pdf.

Annexe A

Mesure de distance

Lorsqu'on souhaite comparer deux vecteurs de caractéristiques issus du module d'extraction de caractéristiques d'un système biométrique, on peut soit effectuer une mesure de similarité (ressemblance), soit une mesure de distance (divergence).

La première catégorie de distances est constituée de distances Euclidiennes et sont définies à partir de la *distance de Minkowski d'ordre p* dans un espace euclidien $\mathbb{R}^N$ (N déterminant la dimension de l'espace euclidien).

Considérons deux vecteurs $X = (x_1, x_2, ..., x_N)$ et $Y = (y_1, y_2, ..., y_N)$, la *distance de Minkowski d'ordre p* notée L_p est définie par :

$$L_p = (\sum_{i=1}^{N} |x_i - y_i|^p)^{1/p}$$

Nous allons présenter quelques mesures de distance dans l'espace original des images puis dans l'*espace de Mahalanobis*.

A.1 Distances Euclidiennes

A.1.1 Distance City Block (L1)

pour $p = 1$, on obtient la distance *City-Block* (ou distance de *Manhattan*) :

$$L_1(x, y) = \sum_{i=1}^{N} |x_i - y_i|$$

A.1.2 Distance Euclidienne (L2)

Pour $p = 2$, on obtient la distance *euclidienne* :

$$L_2(x, y) = \sqrt{\sum_{i=1}^{N} |x_i - y_i|^2}$$

A.2 Distances dans l'Espace de Mahalanobis

A.2.1 De l'espace des images à l'espace de Mahalanobis

Avant de pouvoir effectuer des mesures de distance dans l'espace de Mahalanobis, il est essentiel de bien comprendre comment l'on passe de l'espace des images $\Im_m$ à l'espace de Mahalanobis $\mathcal{E}_{\text{Mah}}$.

En sortie de l'algorithme PCA, nous obtenons des vecteurs propres associés à des valeurs propres (représentant la variance selon chaque dimension). Ces vecteurs propres définissent une rotation vers un espace dont la covariance entre les différentes dimensions est nulle. **L'espace de Mahalanobis est un espace où la variance selon chaque dimension est égale à 1**. On l'obtient à partir de l'espace des images $\Im_m$ divisant chaque vecteur propre par son écart-type correspondant.

Soit u et v deux vecteurs propres de $\Im_m$, issus de l'algorithme PCA, et m et n deux vecteurs de $\mathcal{E}_{\text{Mah}}$.

Soit λ_i les valeurs propres associées aux vecteurs u et v, et σ_i l'écart-type, alors on définit $\lambda_i = {\sigma_i}^2$. Les vecteurs u et v sont reliés aux vecteurs m et n à partir des relations suivantes :

$$m_i = \frac{u_i}{\sigma_i} = \frac{u_i}{\sqrt{\lambda_i}} \text{ et } n_i = \frac{v_i}{\sigma_i} = \frac{v_i}{\sqrt{\lambda_i}}$$

A.2.2 Mahalanobis L1 (MahL1)

Cette distance est exactement la même que la distance *City-Block* sauf que les vecteurs sont projetés dans l'espace de *Mahalanobis*. Ainsi, pour des vecteurs propres u et v de projections respectives m et n sur l'espace de *Mahalanobis*, la distance *Mahalanobis* L1 est définie par :

$$Mah_{L1}(u, v) = \sum_{i=1}^{N} |m_i - n_i|$$

A.2.3 Mahalanobis L2 (MahL2)

Cette distance est identique à la distance euclidienne à part qu'elle est calculée dans l'espace de *Mahalanobis*. Ainsi, pour des vecteurs propres u et v de projections respectives m et n sur l'espace de *Mahalanobis*, la distance *Mahalanobis* L2 est définie par :

$$Mah_{L2}(u, v) = \sqrt{\sum_{i=1}^{N} |m_i - n_i|^2}$$

Par défaut, lorsqu'on parle de distance de Mahalanobis, c'est à cette distance que l'on doit se référer.

A.2.4 Cosinus de Mahalanobis (MahCosine)

Il s'agit tout simplement du cosinus de l'angle entre les vecteurs u et v, une fois qu'ils ont été projetés sur $\mathcal{E}_{\text{Mah}}$ et normalisées par des estimateurs de la variance (figure A.1).

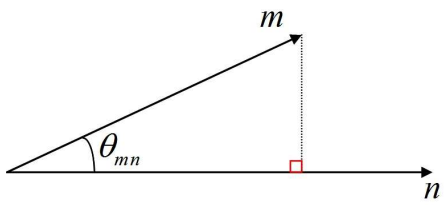


FIGURE A.1 – Les deux vecteurs m et n dans l'espace de Mahalanobis

Nous avons donc par définition :

$$S_{MahCosine}(u, v) = \cos\cos(\theta_{mn})$$

De plus, on peut écrire :

$$\cos(\theta_{mn}) = \frac{|m|\,|n| \times \cos\cos(\theta_{mn})}{|m|\,|n|}$$

D'où la formule finale de la mesure de similarité *MahCosine* :

$$S_{MahCosine}(u, v) = \frac{m.n}{|m|\,|n|}$$

Annexe B

Description de la base de données

Notre base de données se compose de deux grandes parties.

B.1 Première partie

Conditions d'acquisition

En utilisant un appareil-photo fixe, une séquence de 20 images par individu ont été prises. Pendant la séquence, l'individu fait un pas en avant vers l'appareil-photo. Ce mouvement est utilisé pour introduire les variations significatives (échelle) de la tête entre les images d'un même individu.

Il y a environ 0.5 seconde entre les captures successives dans la séquence.

Description de la base de données

- Nombre d'individus : 72
- Résolution d'image : (180×200) Pixel (format de portrait)
- Contient des images d'hommes et de femmes

Variation des images des individus

- Arrière plan : le fond se compose d'un rideau rouge. La variation de fond est provoquée par des ombres pendant que le sujet avance.
- Echelle de la tête : grande variation de l'échelle.
- Tourner la tête, inclinaison : variation mineure de ces derniers attributs.
- Position de visage dans l'image : un certain déplacement.
- Variation d'éclairage d'image : pendant que le sujet avance, des changements significatifs d'éclairage se produisent sur des visages dus à l'arrangement artificiel d'éclairage.
- Variation d'expression : une certaine variation d'expression
- Commentaire additionnel : il n'y a aucune variation du style des cheveux puisque les images ont été capturées dans une seule session.

B.2 Deuxième partie

Conditions d'acquisition

Les sujets se reposent à une distance fixe de l'appareil-photo et sont invités à parler, pendant qu'une séquence d'images est prise. Le discours est employé pour présenter la variation d'expression faciale.

Description de la base de données

- Nombre d'individus : 153
- Résolution d'image : (180×200) Pixel (format de portrait)
- annuaires : femelle (20), mâle (113), malestaff (20)

Variation des images des individus

- Arrière plan : le fond est vert
- Echelle de la tête : aucun
- Tourner la tête, inclinaison : variation mineure de ces derniers attributs.
- Position du visage dans l'image : modifications mineures
- Variation de l'éclairage de l'image : aucun
- Variation d'expression : changements considérables d'expression
- Commentaire additionnel : il n'y a aucune variation du style des cheveux puisque les images ont été capturées dans une seule session.

ملخص :

يتمثل هذا العمل في إنجاز نظام من أجل التعرف على الأوجه يستند على الجمع التعاقبي بين الخوارزمية PCA مع شبكة عصبية اصطناعية متعددة الطبقات.
التصديق على عمل النظام تم بواسطة البرنامج MATLAB. قاعدة البيانات المستخدمة تتكون من 2410 صورة ثابتة لأوجه في حالة أمامية.
من أجل تشكيل النظام على دارة FPGA، استعملنا لغة البرمجة المادية VHDL.
كلمات مفتاحية : التعرف على الأوجه، PCA، شبكة عصبية متعددة الطبقات، FPGA.

Résumé :

Ce travail consiste en la réalisation d'un système pour la reconnaissance de visages basé sur une combinaison séquentielle de l'algorithme PCA avec un réseau de neurones artificiel multicouche.

La validation du fonctionnement du système a été faite par le logiciel Matlab. La base de données utilisée est composée de 2410 images fixes de visages en position frontale.

Dans le but de faire une implémentation sur un circuit FPGA, nous avons utilisé le langage de programmation matérielle VHDL.

Mots-clés : reconnaissance de visages, PCA, réseau de neurones multicouche, FPGA.

Abstract :

This work consists of the realization of a system for face recognition based on a sequential combination of the PCA algorithm with a multi-layer artificial neural network.

The validation of system functioning was made by the Matlab software. The database used is made up of 2410 still images of faces in frontal position.

In order to make an implementation on an FPGA circuit, we used the hardware description language VHDL.

Keywords : Face recognition, PCA, Multi-layer neural network, FPGA.