SOMMAIRE

Projet industriel : système biométrique de reconnaissance

Projet industriel : système biométrique de reconnaissance

Remerciements

Dans l'optique ce projet de recherche et développement, je tiens à remercier toute l'équipe enseignante qui m'a aidé ou soutenu durant ce travail ainsi que les étudiant et tous ceux qui auront manifesté de l'intérêt pour ces travaux.

Je voudrais remercier M.Baloche pour m'avoir fait prendre conscience des risques dans le développement de mon approche initiale dans le contexte de ce projet. Notamment, de m'avoir sensibilisé à mettre en œuvre une démarche progressive et plus pragmatique.

Egalement, je voudrais remercier aussi M.Signolle pour m'avoir éclairé tout au long de ce projet; sans qui je n'aurai pu mettre sur pied une implémentation aussi complète, aussi rapidement sans sacrifier tout ou partie de la profondeur et de la diversité du sujet.

1. Introduction : Motivation et Intérêt du projet.

L'utilisation d'applications d'identification et d'authentification est de plus en plus répandue. Ce fait reflète le besoin pour le système d'information, de reconnaître un usager qui va, une fois authentifié et certifié « de confiance », l'employer à une tâche ou à un service déterminé.

La méthode d'identification la plus répandue est celle du code secret, un numéro à retenir et à restituer, par exemple, lors d'un retrait à une billetterie automatique. Le système d'information du terminal doit s'assurer que le porteur de la carte est bien son propriétaire afin de débiter justement le compte de la bonne personne. Là, il s'agit d'identifier la personne. Plusieurs problématiques subsistent :

- que faire si l'on oublie son code (multiplication des codes secrets à retenir) ?
- que faire si un propriétaire de carte se fait voler son code (et sa carte)? Qu'est-ce qui interdira le voleur d'utiliser carte et code ?
- comment saisir son code hors du terminal sécurisé notamment si on est à distance ? Dans ce cas, comment s'authentifier sans avoir son lecteur de carte personnel ?

Il faudrait concevoir un système « naturel » non-intrusif qui permette à la personne de se faire <u>authentifier et identifier</u> en même temps et d'être le plus passif possible.

<u>Applications possibles de ce système et de ce système dérivé (classé par ordre croissant de remodélisation nécessaire) :</u>

- Sécurisation de transactions (remplacement du PIN, signature électronique pour document)
- Autorisation d'accès (lieux, produits, services)
- Identification de forme (tri par catégorie, distribution d'attributs)
- Intelligence artificielle (émergence de forme, conscience artificielle)

Ce genre de système existent (ou sont « en passe » d'exister) et sont à la pointe du progrès technologique dans le domaine tout jeune de la bioinformatique. Des années de recherches de plusieurs laboratoires, groupes et consortiums, des milliers de personnes ont fourni des solutions et des méthodes de plus en plus efficace. Mon but dans ce projet n'est pas d'en rajouter, il est d'étudier ce domaine passionnant, tout au plus d'implementer un système fiable tout en préservant la relative simplicité des calculs statistiques. Ce projet est orienté vers l'industrialisation et décrit des procédés et techniques que je tenterais de documenter en détail aussi clairement que possible.

2. Vue d'ensemble du projet.

J'ai abordé ce projet industriel avec les méthodes de travail utilisées dans la recherche et le développement. J'ai donc tout naturellement commencé par faire un état de l'art. Suite à cet état de l'art des solutions existantes d'authentifications et après concertation avec M. Balloche et M. Signolle, il nous est apparu que la méthode EigenFace était une approche intéressante pour plusieurs raisons :

- C'est une des méthodes les plus efficace de reconnaissance.
- Il est relativement rapide à implémenter (en rapport à des analyses neuronales RBF et autres)
- Cette approche connaît un succès croissant dans les laboratoires.

La méthode EigenFace est une méthode qui vise à dégager des EigenFaces. Ce sont les visages synthétiques contenant les caractères biométriques représentatifs de la forme à reconnaître, ils sont calculés depuis les différentes photographies numériques d'une banque de données. Ces EigenFaces, sous forme de clé publique, sont la représentation caractéristique d'un visage, unique... une signature inimitable.

Un scénario réponse à la problèmatique soulevée :

On peut calculer une clé privée d'un nouveau visage d'une personne dans le terminal A. Au moyen de la clé publique, calculer une clé privée qui, petite, peut être stockée dans la mémoire d'une carte à puce. Cette personne portera cette carte, et lorsqu'elle viendra se faire identifier auprès du terminal B, elle la glissera dans un lecteur de carte du terminal et une photo sera prise d'elle. Le terminal C, à partir de la clé publique, d'une autre photo et de la clé privée contenue dans la puce de la carte, pourra estimer le plus justement possible si la personne qu'il a pris en photo, et bien le porteur véritable de la carte.

Pour le projet, je dispose de terminaux X dans un environnement SCO Unix muni d'Octave (Le projet qui donna naissance à sa version commerciale Matlab) et du logiciel de traitement d'image Jasc Paint Shop Pro, de 3 heures par semaine pour environ 14 semaines de projet soit 42 heures au total.

Ce rapport d'activité décrit bien sûr le protocole pour reproduire l'implémentatoin décrite briévement ci-dessus. Mais il donne surtout une vision plus ambitieuse, avec la démonstration continuelle de sa faisabilite et son efficacité. Je propose une architecture, un système d'information qui peut supporter une charge « industrielle » et non plus qu'expérimentale. Les difficultés soulevées ainsi que les solutions innovantes proposées sont décrits dans ce document.

En annexe, vous retrouverez l'étude de l'existant ou état de l'art ainsi que les raisons de choix de l'approche par Eigenface.

3. Acquisition : se constituer une banque d'images.

L'acquisition comprend la prise de la photo, et il s'agit également de faire toutes les opérations de retouche d'image afin d'atteindre les spécifications de la norme décrite en page suivante.

J'ai donc récupéré quelques photos du trombinoscope des étudiants en licence M.C.A, transformées cela constituera ma banque de donnée du terminal certifieur. L'acquisition des photos est donc déjà faite et le prétraitement des photos est anuel, cela aura l'avantage de se rendre compte empiriquement des normes à définir pour atteindre des résultats jugés corrects. Cela afin de spécifier un futur traitement automatisé.

Le format de ces fichiers sera convertit en ppm ascii(portable pixel map). Ce format s'il est peu compact présente un intérêt majeur, il reste lisible dans un éditeur de fichier basique et manipulable facilement. Il se présente de la façon suivante :

```
P3                                  |format du fichier
# Created by Paint Shop Pro 7       |commentaire divers
122 153                             |taille de l'image (col,lig)
255                                 |valeur maximal d'un pixel
185 176 175                         |pixel = triplet RougeVertBleu
```

Les images doivent, afin de constituer un ensemble homogène, répondre à une norme fixant des caractéristiques afin de faciliter les traitements. Ces normes doivent être appliquées lors de l'acquisition de l'image.

Spécifications de la norme:

- Taille de l'image : environ 100 par 100 pixels pour le cadre, taille du visage environ 70 par 70.
- Profondeur de couleur : 8 bits 256 teintes/couleurs maximum en niveau de gris.
- Zone visage : un cadre de sélection afin de ne conserver que la zone rectangulaire depuis les sourcils (inclus) et du bord des yeux jusqu'au bas du menton.
- Moyenne Luminosité : 127
- Histogramme des teintes : égalisée .
- Neutralité de l'expression du visage : type photo identité
- Source lumineuse : constante, orientée à 10-11h ou à 1-2h de face à l'objet.

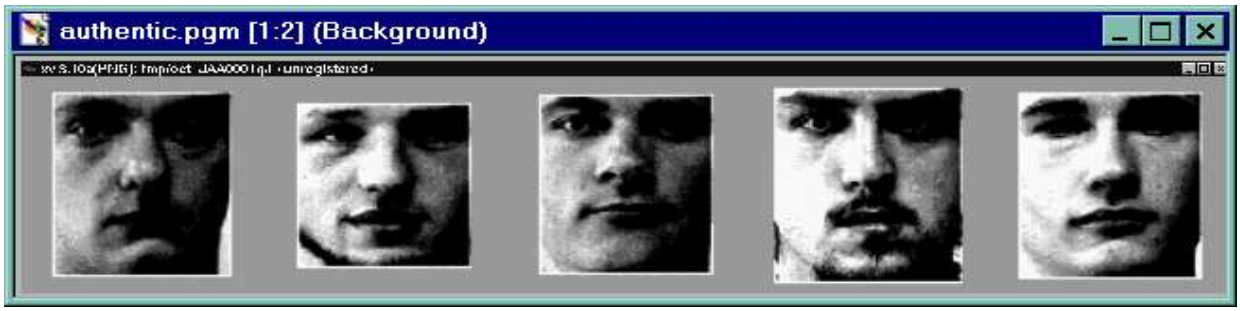
L'avantage d'une profondeur de couleurs 8 bits est de pouvoir travailler en niveau de gris afin de faire apparaître les reliefs du visage (les teintes claires étant les zones éclairées et les teintes sombres étant les zones d'ombre).

Donc je vais passer d'un format .ppm au format .pgm (Portable Grey Map). Les trois valeurs de teintes (RVB) seront donc les mêmes. Après enregistrement dans un fichier au format .pgm, voici l'image (tronquée) :

```
P3
# Created by Paint Shop Pro 7
76 86
255
158 158 158 125 125 125 224 224 224 52 52 52 38 38 38 38 38 38 38 38 38
```

Cette image, issue du trombinoscope, déjà retravaillée et compressée en différents formats n'est certes pas idéale, et les traitements d'images seront moins efficaces. A l'avenir les traitements relatifs au processus d'acquisition de l'image devront aborder la réduction de bruits, et l'homogénéisation du contraste/luminosité (éclairages) sur des images « analogiques » (sans antécédent de numérisation) et non compressées.

Une des difficulté majeure de l'acquisition (automatique ou non) réside dans les choix normatifs. Par exemple, le critère proportion laissé pour l'heure à l'appréciation humaine (lors de la prise de vue et du découpage du cadre), devrait être calculée. En l'état actuel, toutes les images de 100x100 sont centrée dans leur cadre. Voici la « galerie » d'image, que je retravaillé avec les normes:



On le remarque immédiatement n'est-ce pas ? Si l'on superpose les images les unes sur les autres, les yeux, le nez et la bouche ne coïncideront pas parfaitement. C'est normal, tout le monde n'a pas le même faciès et heureusement sans quoi, la vie ne serait pas aussi riche mais aussi, techniquement parlant, ce type de reconnaissance tomberait à l'eau. La perfection n'est pas requise ici, il suffit que les proportions des visages les uns par rapport aux autres soient, au mieux, identiques.

Est-ce que le calcul du critère proportion (qui serait un agrégat de calculs en fait) par exemple de la distance entre les yeux est une nécessité et si oui est-ce faisable ?

Oui et oui, mais cette exemple de calcul de la distance entre les deux yeux n'est pas si simple : une solution consisterait à détecter les yeux en multipliant par personne ces images. Il faudrait faire une photo les yeux fermés puis une seconde, immédiate avec les yeux ouverts. Un traitement discriminant par différence m'informerai sur la position des yeux dans l'image afin de calculer cette distance. Cette distance rapportée à celle du cadre donnerait un facteur « de zoom ». Finalement, celui-ci devrait être le même (à peu prés) pour chaque photo.

Seulement ce facteur ne serait pas vraiment un zoom sans que l'on calcule une distance verticale (par exemple distance sourcils - menton) que l'on rapporterait à la dimension verticale du cadre. Le rapport de ces deux calculs de distance, que l'on pourrait appeler cadrage, serait comparé à un autre rapport fixé par la norme d'acquisition. Le résultat de cette comparaison, le zoom proprement dit, devrait être nul.

Le rapport (X'/X) du visage numéro 2 sera visiblement plus grand que celui du visage numéro 4, une alerte programmée pourrait alors très bien le signaler à l'usager. Pour ce qui est du rapport (Y'/Y), le choix de la distance (sourcil – menton) n'est pas très probant car on mesure presque le cadre de l'image lui-même : il faudrait mieux ne pas l'intégrer pour définir le cadrage.

Mais on pourrait très bien, rajouter d'autre distances ou mesures, et déjà remplacer la distance menton – sourcil, par la distance lèvre supérieure et bout du nez. Il faudrait pas que la personne ne fasse pas la moue (il y a un critère de neutralité d'expression) mais la neutralité du visage est très relative d'un individu à un autre **… La distance commissure des lèvres et bord de l'œil semble plus approprié.**

Il faudrait encore détecter la commissure des lèvres, bouche ouverte – bouche fermée pourrait convenir ? Quoiqu'il en soit, pour n critères entrant en compte dans l'agrégat cadrage, il faudrait faire 2^n clichés.

Pour des test expérimentaux, si ces deux critères pouvaient suffire en sachant qu'on veut une banque de donnée de dix visages, il faudrait déjà 40 clichés ! et 40 traitements manuel d'images !

$$\text{nombre de cliché nécessaire} = 2^{\mathbf{nbcritère}} \text{ X nombre de personne}$$

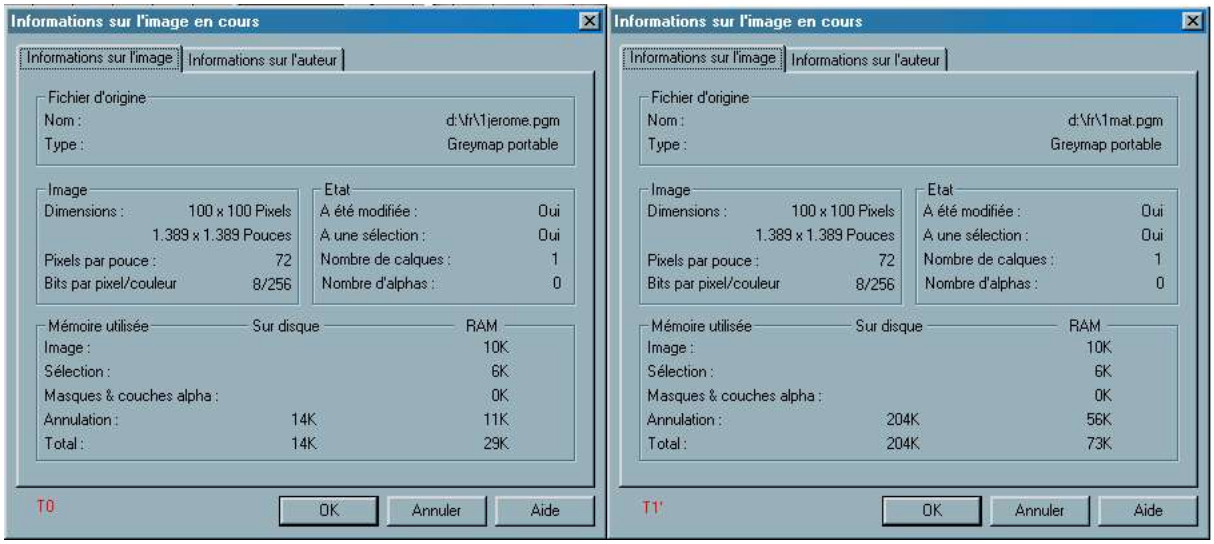
Un tel dispositif pourrait signaler des photos hors norme grâce à des critères mesurés C_i' et leur base normale C_i .

$$\text{Cadrage photo} = (C_1'/\ C_1) / (C_2'/\ C_2) / (C_3'/\ C_3) \ldots / \ \ldots (C_n'/\ C_n) \ \overset{?}{=} \text{Cadrage norme}$$

Nécessairement, si n est élevé un étalonnage de la norme devra être fait sur la base d'une moyenne de Cadrage des photos de la banque d'images. Ceci rajouterai des calculs et incertitudes qui se rajouterai à celles issues des critères, telles une commissure mal dessinée, etc ..

Je compte m'affranchir de ces difficultés comme je l'ai abordé dans la vision globale du projet, acquisition reste purement manuelle. Elle pose en elle-même déjà suffisamment de difficulté et de temps de travail.

Il existent d'autres moments où l'acquisition est nécessaire c'est lors de la personnalisation et de l'authentification . Admettons, j'insère une nouvelle image dans le système :



Il faut prendre des précautions d'acquisition lors de l'ajout dans la banque de donnée tout autant qu'à sa constitution . Tout particulièrement pour un traitement manuel, ci-dessus T0 est une image normée typique, un visage de la banque de données.

T1' est une autre image, issue du terminal de personnalisation (image authentique) ou du terminal d'authentification (image à authentifier), après la même normalisation que celle de la constitution de la banque de données.

Remarque : il y a une information à ne pas mal interpréter, la taille de l'image T1' est grossie à cause de la taille de son image cache : elle est en cours de modification (lors de la capture d'écran).

Comme je procède manuellement, une légère différence de taille peut-être due au traitement [1] qui est passage en 8 bits de profondeur de couleur (... en niveau de gris), suivi d'un redimensionnement des images selon un pourcentage [2], une égalisation d'histogramme, d'un pourcentage du luminosité [3], finissant par la sauvegarde dans un format .pgm.

Exemple de calcul (de dimension) par proportionnalité :

T1 : Taille de la photo d'acquisition : x : 640 y : 480
T0 : Taille des photos de la banque : x :100 y :100

Coeff. x = 100/640 soit 16% Coeff. y = 100/480 soit 21%

T1' : Taille de la photo traitée par réduction : x : 100 y :100

Nous avons vu que l'acquisition est une phase délicate dans ce projet et que j'ai trouvé un outil secourable en la disposition (de fait). Pour clore cette partie importante il semble essentiel, incontournable de passer en revue les « bonnes pratiques » tant au niveau technique qu'humain.

[1] Diminution du nombre de couleur : Palette : coupe médiane optimisée ; méthode : diffusion, puis niveau de gris
[2] Dimensionner : Pourcentage de la taille original (pourcentage approximé) en X et Y ; type de redimensionnement : taille intelligente
[3] Luminosité : Pourcentage de luminosité à appliquer linéairement par rapport aux images en banque.

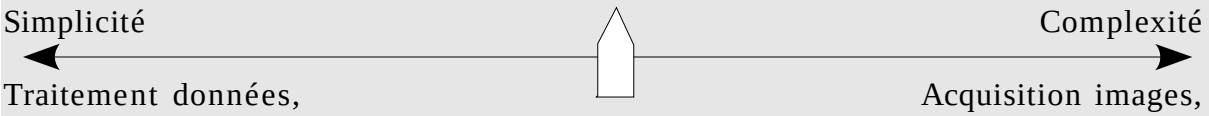
Considérations sur le rôle de la banque d'images et sur l'éthique de la biométrique

C'est un élément très important du système car il inférera directement sur la fiabilité de l'authentification. La banque de d'image doit définir au mieux la forme à reconnaître afin d'éviter les futures ambiguïtés qui pourrait naître d'une confusion de forme.

En outre, on pourrait prendre en photo et de mettre en banque tous les visages humains. Bien sûr, on se constituera un échantillon de cette population ! Il faut un échantillon adapté : ni trop petit, afin d'éviter une reconnaissance parcellaire du visage ; ni trop gros, afin de ne pas avoir de problème de stockage de l'information

Le cadre normé de sélection du visage lors de l'acquisition élude les problèmes pouvant naître d'élément « variant fréquemment » : coiffure, bijoux, chapeau, etc.. . C'est en cela qu'on aperçoit que le dispositif normatif permet de faciliter ; le traitement des données et l'interprétation mais complique rapidement l'acquisition en elle-même et le traitement de l'image... et inversement : Un ajustement fin est nécessaire.

Implémntation d'un ajustement moyen de la complexité entre la constitution des données (collecte des images) et la construction d'un résultat au moyen d'un dipositif normatif.



Traitement données, Interprétation données : *Construction d'un résultat.*	Acquisition images, Traitement images : *Constitution des données.*
Faible quantité/qualité des critères normatifs	Forte quantité/qualité des critères normatifs

Pour aller jusqu'au bout de l'idée, il faudrait établir une moyenne, sur différents système, afin de déterminer une résolution normée permettant cette finesse dans l'ajustement.

Il faut avoir une banque de photo homogène mais variée. On veut pouvoir reconnaître tout les différentes formes de visage, d'yeux, de nez, etc.. il faut introduire par conséquent des formes de visage, d'yeux, de nez différents. Il faut se constituer un échantillon représentatif de la forme à reconnaître.

Dans de nombreuses études, on aborde le faux problème (donc insoluble) de quantification obscure de la proportion de visage à peaux sombres/clairs. Celles-ci voulant peut-être répondre à une véritable question : comment faire pour ne pas « faire chuter » son échantillon ?

Il faut qu'il réponde à ces critères :

Critères de l'échantillon :

- Homogène mais;
- Varié mais;
- Représentatif, médian mais de;
- Taille adaptée aux capacités de stockage

de la cible et au besoin de résolution de l'authentification.

Un visage de complexion très pale ou très foncé ne pose pas plus de problème qu'un visage au nez déformé ou droit ... En partant du principe que peu de visage sont très déformé, très coloré, très en relief ou très spécifique pour généraliser, il a sa place dans cet échantillon sans faire chuter la « moyenne ».

Même en raisonnant à l'inverse, et s'il y en avait beaucoup avec une spécificité particulière, l'homogénéité n'en serait non plus mis à mal puisque ces visages deviendraient une base de définition, rééquilibrant le système vers un renouveau représentation (la majorité sinon l'importance en nombre faisant « loi » dans une « moyenne »).

Ceci pour dire également, qu'il faut que le système informatique s'adapte à la condition humaine et non l'inverse. C'est un mal emploi que de la dégrader pour des considérations techniques, tâchons alors de répondre de la diversité humaine avec la diversité des outils techniques dont nous disposons, pouvu que nous réussissions à les mettre en oeuvre.

Comme il est vrai qu'un échantillon doit être durable, il sera mal venu de devoir repersonnaliser des milliers de cartes pour des raisons évidentes de coûts. Justemment, la population que représente l'échantillon ne varie pas tant que ça. Cette opération, si elle devait être produite fréquemment, ne trouverait d'autre intérêt que dans l'assurance de la sécurité du système par peur de piratage (changement, mise à jour des clefs).

Il faut retenir que grosso modo, la spécificité de ce visage peut être à l'origine de la création d'un critère biométrique servant à définir, de façon plus complète avec une résolution plus fine, un autre visage, même bien différent : c'est ce qui est recherché !

4. Pré traitement : Préparation des données

En l'état, nous disposons d'une banque d'image. Toutes les photos normalisées au format .pgm sont prêtes à être « injectées » dans le système afin de générer un clé publique. Mais cette injection ne peut pas se faire directement sans transformation de ces images.

L'objectif est de modifier le fichier visage .pgm, afin qu'il puisse être interprété par Octave. Le bon choix du format facilite déjà la récupération des données sous Octave. Sous Octave tout est vecteur ou matrice, et scalaire.

En définitive, seulement trois variables caractérisent une image : sa taille (un vecteur X,Y), le nombre maximal de couleur (un scalaire ncol) et le pixmap, c'est-à-dire la représentation de l'image elle-même (un vecteur de la taille de l'image X,Y). Idéal, c'est qu'Octave lors du chargement de l'image, récupère tout ceci depuis un seul fichier. Seulement le fichier .pgm devra être modifié car brut il est incompréhensible pour la fonction de chargement d'image d'Octave **load.**

Ce qu'Octave ne peut pas comprendre	**Ce qu'Octave interprète**
P3 # Created by Paint Shop Pro 7 100 100 255 158 158 158 158 158 … Fichier 1.pgm	#P3 #This file has been generated by prepare.sh for Octave load function # name: dims # type: matrix # rows: 1 # columns: 2 100 100 # name: ncol # type: scalar 255 # name: X # type: matrix # rows: 100 # columns: 100 158 158 158 158 158 … Fichier 1.pgm.oct

Il a donc fallu que j'implémente un script shell (prepare.sh) qui lancé sur le fichier 1.pgm me donne un fichier de données initiale 1.pgm.oct

Voici le code de prepare.sh, il peut être lancé à la main …

```
#!/bin/bash

if [ -z "$1" ]
then
  echo "Usage: `basename $0` filename-to-convert"
  exit 1
fi
NEWFILENAME="$1.oct"
FILENAME="$1"

echo ".PGM File : $FILENAME"
echo ".PGM.OCT File : $NEWFILENAME"

NBLINES=`wc -l $FILENAME | cut -f1 -d' '`
X=`head -n4 $FILENAME | tail -n2 | head -n1 | cut -f1 -d' ' `
Y=`head -n4 $FILENAME | tail -n2 | head -n1 | cut -f2 -d' ' `
NCOL=`head -n4 $FILENAME | tail -n2 | tail -n1`

echo "#P3" >$NEWFILENAME
echo "#This file has been generated by $0 for Octave load
function"  >>$NEWFILENAME

echo "# name: dims"  >>$NEWFILENAME
echo "# type: matrix"  >>$NEWFILENAME
echo "# rows: 1"  >>$NEWFILENAME
echo "# columns: 2"  >>$NEWFILENAME
echo "$X $Y" >>$NEWFILENAME

echo "# name: ncol" >>$NEWFILENAME
echo "# type: scalar" >>$NEWFILENAME
echo "$NCOL" >>$NEWFILENAME

echo "# name: X"  >>$NEWFILENAME
echo "# type: matrix"  >>$NEWFILENAME
echo "# rows: $X"  >>$NEWFILENAME
echo "# columns: $Y"  >>$NEWFILENAME

rm -f temp
touch temp
vi "$FILENAME" <<!
4dd
:w! temp
:q!
!
echo "Header: 14 lines written."
echo "Image : $NBLINES lines written."

cat temp  >>$NEWFILENAME
rm -f temp
exit 0
```


Plus intéressant, Octave peut grâce à une commande 'invoquer le shell' lancer ce script pour préparer toutes les images d'un répertoire par exemple.

Alors dans un fichier script shell, je placerai autant d'appels au script ci-dessus afin de convertir autant d'images que l'on veut voir figurer dans la banque de données. Ceci générera n photo-script de type .pgm.oct qui pourront être assimilé par la fonction load d'Octave.

TraitementPhoto.sh

```
Echo « #Erreurs générées par prepare.sh » > err_prep.log
Echo « # lors de la préparation des données » >> err_prep.log
Echo « Début de préparation des données… »
Sh prepare.sh 1.pgm 1>/dev/null 2>> err_prep.log
Sh prepare.sh 2.pgm 1>/dev/null 2>> err_prep.log
Sh prepare.sh 3.pgm 1>/dev/null 2>> err_prep.log
.
.
Sh prepare.sh n.pgm 1>/dev/null 2>> err_prep.log
Echo « Fin de préparation des données… »
```

Voilà, désormais la partie à proprement parler d'acquisition des données et d'injection dans le système est terminé.

Il va falloir manipuler et faire les opérations nécessaires sur ces données et cela requière une méthode, - un programme, - mais surtout une organisation systèmique modulaire & sécurisée, répondant à un besoin spécifique. Il faut que ceci reste assez générique car c'est susceptible d'être porté ou dérivé.

En annexe, vous retrouverez les scripts complets de préparation des données prepare.sh et traitphoto.sh.

5. Proposition d'implémentation d'un système de reconnaissance.

Trois sous-systèmes, physiquement trois terminaux ont été dégagés suite à la considération des cas d'utilisations, ils sont basés sur une architecture standard à crytographie asymétrique. L'idée vient de l'intuition et de la prise de conscience que je demande à ce système une sorte de dialogue notamment entre l'entité qui authentifiera et l'entité qui aura à s'authentifier; et cela sans aborder le rapporchement évident avec le dialgoue IFD - ICC

Il s'agissait également de pouvoir bénéficier des propriétés de sécurité indispensable des systèmes de cryptographie qui offrent des dispositions futures d'intégration.

- Un terminal de certifieur :

C'est la tierce partie de confiance, qui traite les images selon les normes en vigueur et calcule par traitement eigenface, une clé publique. Celle-ci est mise à disposition de façon sécurisée au terminaux de personnalisation et d'authentification qui se sont « abonnées » du système et réseau de reconnaissance faciale.

- Un terminal de personnalisation :

C'est la partie qui dialogue avec le terminal d'authentification indirectement par le biais d'une carte à puce qu'elle émet et personnalise au préalable avec l'autorisation du terminal certifieur. Ils génèrent à eux deux un couple clé publique – clé privée, cette dernière est stockée dans la puce de la carte, mise à la dispostion d'un usager.

- Un terminal d'authentification :

C'est l'autre partie qui dialogue indirectement avec le terminal de personnalisation. Elle se compose du terminal proprement dit de la carte à puce. Ces deux entités réalisent l'authentification.

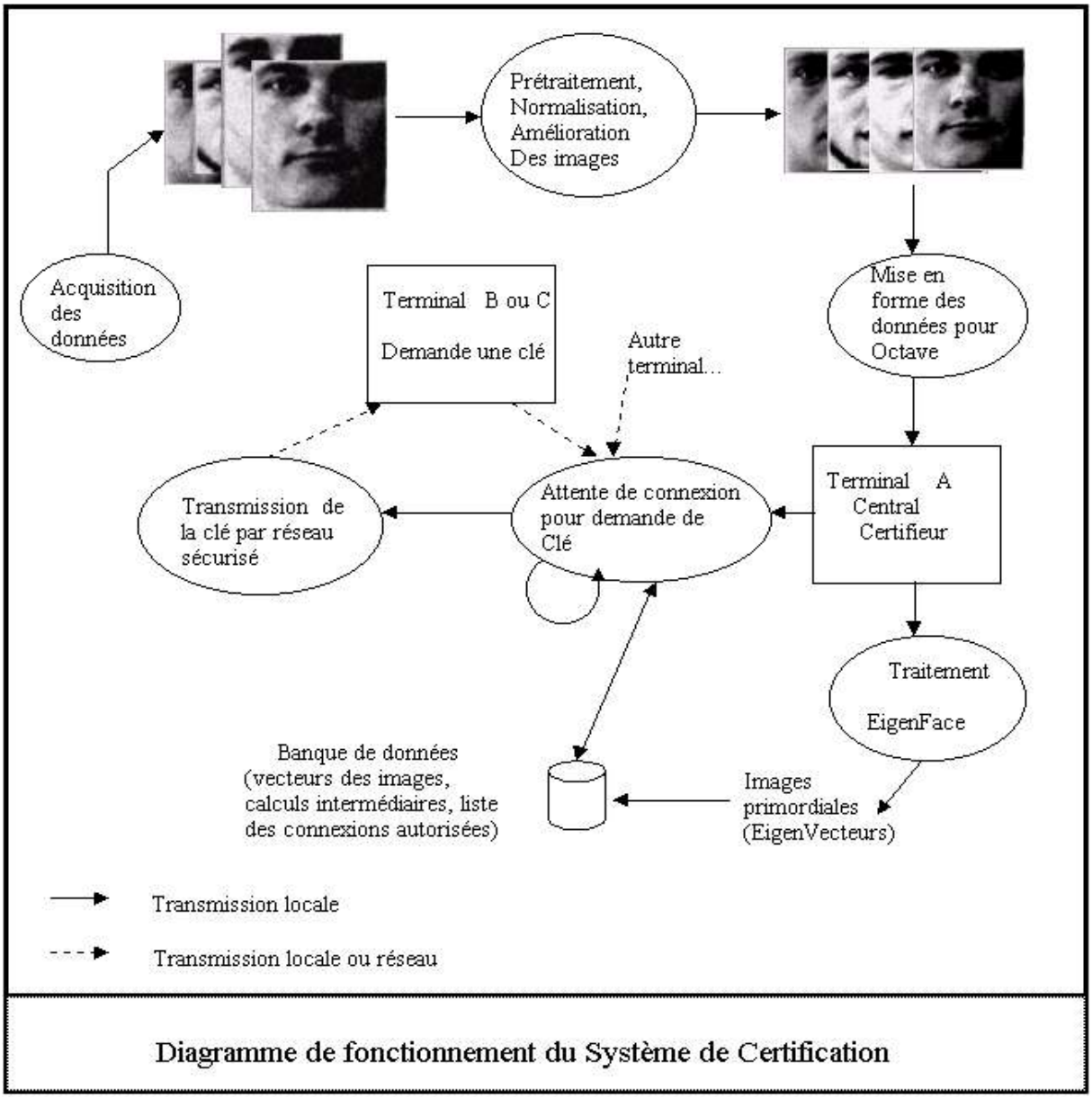


Diagramme de fonctionnement du Système de Certification

Dans ce premier diagramme, l'usager du système ou l'utilisateur est, typiquement, un super administrateur ; par exemple une société qui développerait et distribuerait ce système d'authentification. A l'initialisation de ce système, on insère toutes les photos et on calcule les eigenvecteurs. Ces résultats ainsi que les calculs intermédiaires peuvent être stockés dans la banque de données.

Note sur la confidentialité : Les eigenvecteurs représentent, caractérisent les photos en une forme algébrique matricielle. Les images modifiées sont méconnaissable par rapport aux initiales . Seuls ces modifications sont utiles lors de l'authentification. Les images initiales sont des données uniquement lue. Ensuite, elles peuvent (doivent ?) être détruites après cette phase d'initialisation .

Projet industriel : système biométrique de reconnaissance

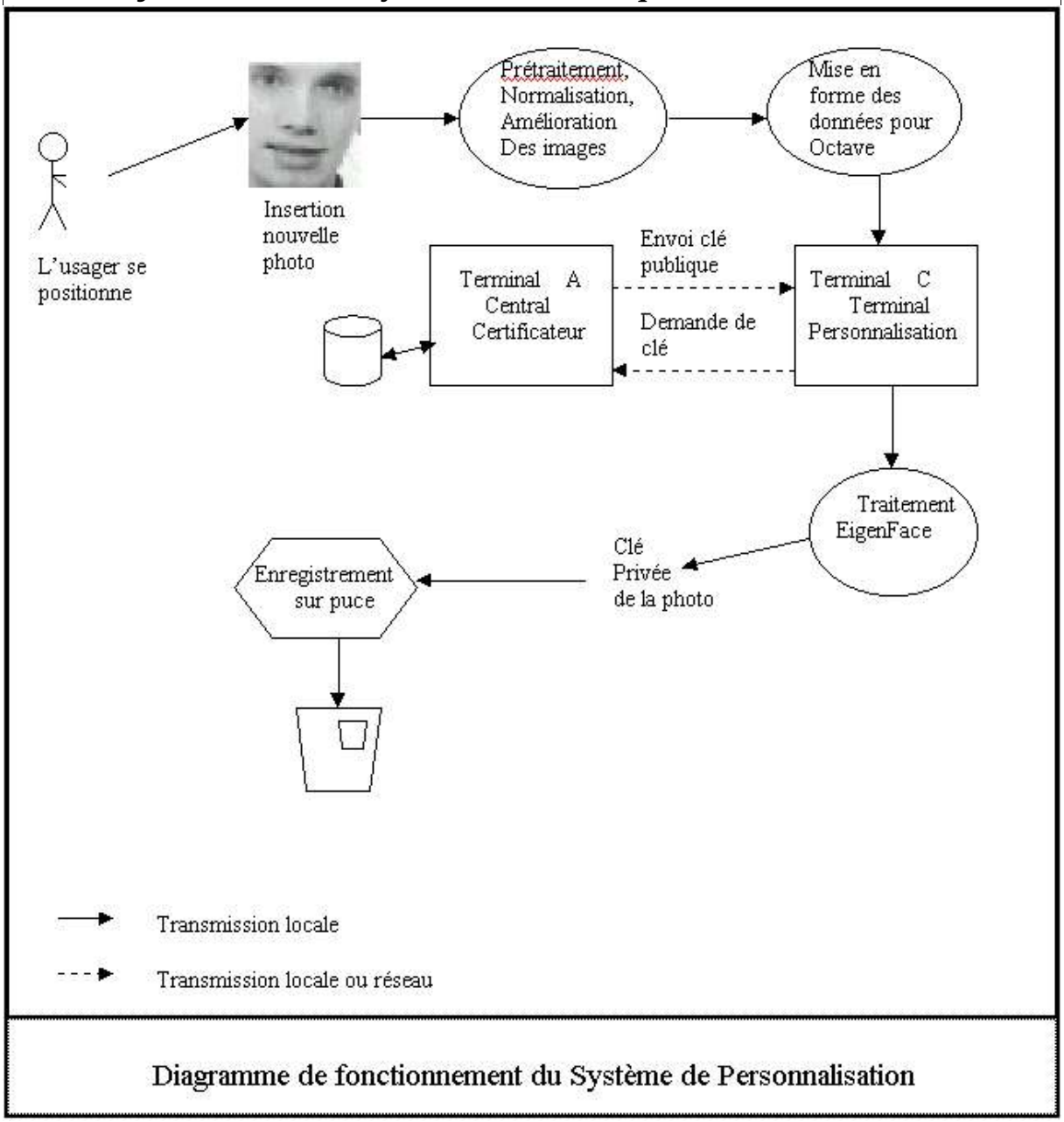


Diagramme de fonctionnement du Système de Personnalisation

Dans ce deuxième diagramme, l'usager du système ou l'utilisateur est, typiquement, un administrateur qui commence par initialiser le système de personnalisation ; par exemple une société de sécurité qui aurait loué ce système à la société certificatrice. Puis à la demande (d'un utilisateur final), il crée des usagers du système d'authentification, des clés privées , qu'il enregistre sur des cartes à puces.

Note sur la confidentialité : Idem que pour le système précédent, la nouvelle photo une fois numérisée peut-être (doit être ?) détruite.

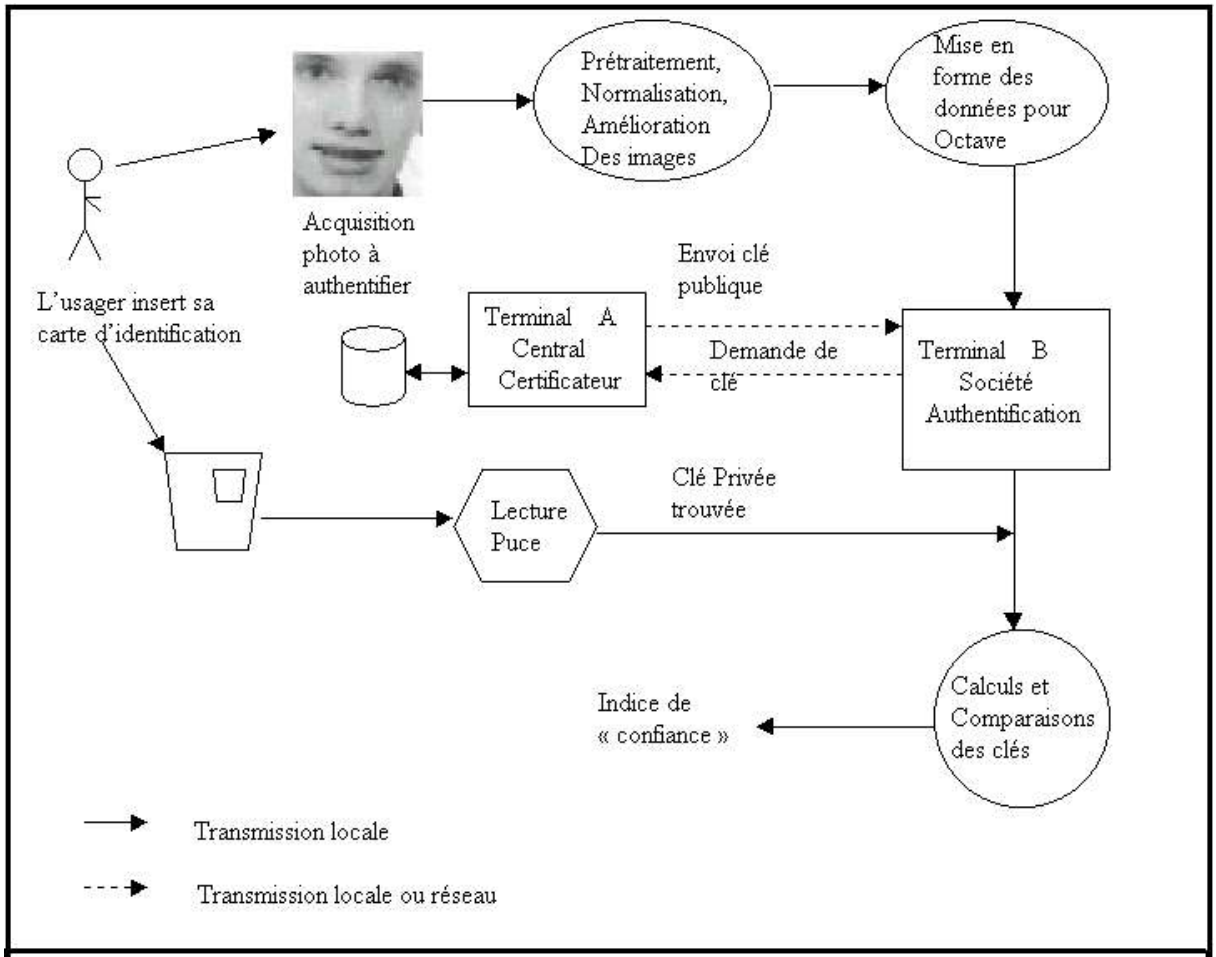


Diagramme de fonctionnement du Système d'authentification

Dans le troisième diagramme correspondant à un deuxième cas d'utilisation, l'usager est typiquement un utilisateur sans droit particulier, qui serait un client de la société de sécurité. Il insert sa carte identification et le lecteur récupère la signature de la puce. Après traitement, et si l'indice de confiance est suffisamment élevé au regard de ce que la personnalisation (de ce terminal) aura défini, l'usager aura accès ou non au service qu'il demande.

Note sur la sécurité du système : Dans l'idéal c'est la carte qui devrait faire une demande de clé publique et de faire le test en interne, c'est-à-dire sans que la clé privée qu'elle stocke n'est le besoin d'en être extraite. Ainsi la carte munie des 2 clés pourrait renvoyer l'indice de confiance, le terminal ne ferait (impératif de sécurité de la monétique) que du calcul de présentation des données. Les calculs de clé et éventuels calculs cryptographiques doivent être accomplis à la discrétion de la carte seule. Ils se doivent donc de ne pas être trop complexe sous peine d'étendre le temps de calcul et donc le temps de réponse de la carte de façon inacceptable (normalement le temps d'attente utilisateur doit être de quelques secondes) .

Difficultés/solutions de cette architecutre dans la contexte du projet.

En raison du temps dont je dispose, le système va être simplifié le plus possible sans modifier l'essentiel du fonctionnement et l'organisation du système que je propose.

Les trois terminaux A, B, C seront les mêmes et tout les traitements seront locaux, je m'attacherai simplement à effacer l'environnement de l'unique terminal entre chaque processus devant s'exécuter sur ces trois terminaux différents.

Enfin, la clé privée restera stockée pour l'heure sur le terminal mais l'implémentation prévoira une place pour deux codes, l'enregistrement en carte de la clé privée et la lecture en carte de cette clé (dans les diagrammes ces « emplacements » sont représentés par des hexagones).

Maintenant que l'architecture du sytème et le rôle de ses organes sont définis, que les difficultés dues au contexte de projet sont solutionnées, nous pouvons considérer cette première conception comme terminée (nous verrons dans la partie bilan qu'une phase de conception secondaire sera alors envisageable). Le travail d'implémentaton peut commencer.

6. Traitement des données.

Implémentation du terminal certifieur

Le terminal certifieur est celui qui constitue une banque d'image, son rôle est de de générer une sorte de clé publique.

Le but est d'appeler la fonction native d'Octave Eigenface eig(v) sur ce vecteur v. C'est elle qui donne les EigenValues, les proportions de caractères biométriques de tous les visages les uns pour rapport aux autres.

Ces caractères sont basés sur les covariances des teintes des pixels des images les uns par rapport au autres. De telle sorte qu'en niveau de gris, des reliefs (zone de teintes) communs / dominants / émergeant de tous les visages décrivent une forme caractéristique, une définition d'un visage humain. Dès lors, je caractériserai les visages d'un échantillon représentatif de visage d'humains avec cette définition, je projetterai mon échantillon de visages dans l'espace de cette définition. Cela pour obtenir des visages synthétiques, projetés qui auront été « remodelés » par la forme caractéristique commune.

Maintenant, il reste à charger ces scripts-photo dans Octave. A cet effet, j'ai crée un fonction fr_load (stocké dans un fichier fr_load.m) qui permet de les mettre en forme de vecteurs colonne, condition facilitant le traitement par EigenFace. D'ailleurs, cette fonction qui recouvre la prise de vue ainsi que l'injection dans le système, sert également aux autres terminal, elle est commune .

Note : Toutes les fonctions Octave que je créerai dans le cadre de ces travaux sur la reconnaissance faciale seront de forme normée fr_nomdefonction().

Voici en l'état, le code de ma fonction fr_load :

```
function [y,depth,dim,mmm,nbfoto] = fr_load (C_Taille)
%
% Fr_load admet la taille X*Y de l'image en paramètre.
%  Elle recherche tout les fichier .pgm.oct du répertoire
%  courant. Elle met en forme toutes les images au format
%  vecteur colonne. Elle vérifie également la cohérence
%  des données de ces images.
%
%         nbfoto : nombre de photo du répertoire courant
%         y : Banque de données vecteur colonne image de taille
%               [C_Taille,nbfoto]
%         depth : profondeur de couleur des images chargées
%         dim   : dimension [X,Y] d'une image
%         mmm : vecteur contenant des données calculée sur y
%               [Teinte min, max , moyenne]
%
%
```

```
          f=popen ("ls -1 *.pgm.oct | wc -l","r");
          n=fscanf(f,"%d");nbfoto=n;
          printf (" %d images .pgm.oct trouvées \n",n);fclose(f);
          f=popen("ls -1 *.pgm.oct","r");

           for i=1:n
                 file=fscanf(f,"%s",1);
                 cmd=sprintf("load -force %s",file);file
                 eval(cmd);

                  if (dims(1)*dims(2)!=C_Taille)
                       printf("%s: La taille de l'image [%i,%i] et le
paramétre \nde la fonction %i doivent correspondre !\n",file,dims(1),dims
(2),C_Taille);
                       clear;
                 end

                 X=reshape(X,C_Taille,1);
                 depth=ncol;
                 dim=dims;
                       if !exist("y")
                             y=X;
                       else
                             y=[y,X];
                       end
           end
          mmm=[min(min(y)),max(max(y)),mean(mean(y))];
          if (  ((mmm(1)<0) || (mmm(1)>255))  ||  ((mmm(2)<0) || (mmm(2)
>255)) || ((mmm(2)<0) || (mmm(2)>255))   )
           printf("min: %i , max: %i , moyenne: %i valeur hors norme !
",mmm(1),mmm(2),mmm(3));
           clear;
          end

          s=size(y);

          if (s(1)!=C_Taille || s(2)!=n)
           printf ("Erreur lors du chargement ...");clear;
          end

          fclose (f);
```

A la sortie de cette fonction, en supposant qu'il y ait 5 photos .pgm.oct dans mon répertoire courant de taille 100x100 chacune, voici un exemple d'appel de ma fonction et le contenu de mon environnement sous Octave :

```
         # octave
         GNU Octave, version 2.0.14.93 (i486-pc-sysv5).
         Copyright (C) 1996, 1997, 1998, 1999 John W. Eaton.
         This is free software with ABSOLUTELY NO WARRANTY.
         For details, type `warranty'.

         octave:1> [y,depth,dim,mmm,nbfoto] = fr_load (10000);
          5 images .pgm.oct trouvées
         file = 1alban.pgm.oct
         file = 1etienne.pgm.oct
         file = 1jerome.pgm.oct
```

```
file = 1pean.pgm.oct
file = 1seb.pgm.oct
octave:2> who

*** currently compiled functions:

columns  fr_load  isempty  mean     printf   reshape  rows
strcmp

*** local user variables:

depth   dim     mmm     nbfoto  y
octave:3> size (y)
ans =

  10000      5

octave:4> nbfoto
nbfoto = 5
```

Pour appliquer la méthode eig (Traitement EigenFace), il faut calculer les variances des colonnes les unes par rapport aux autres, pixel à pixel : la fonction covariance est utile ici :

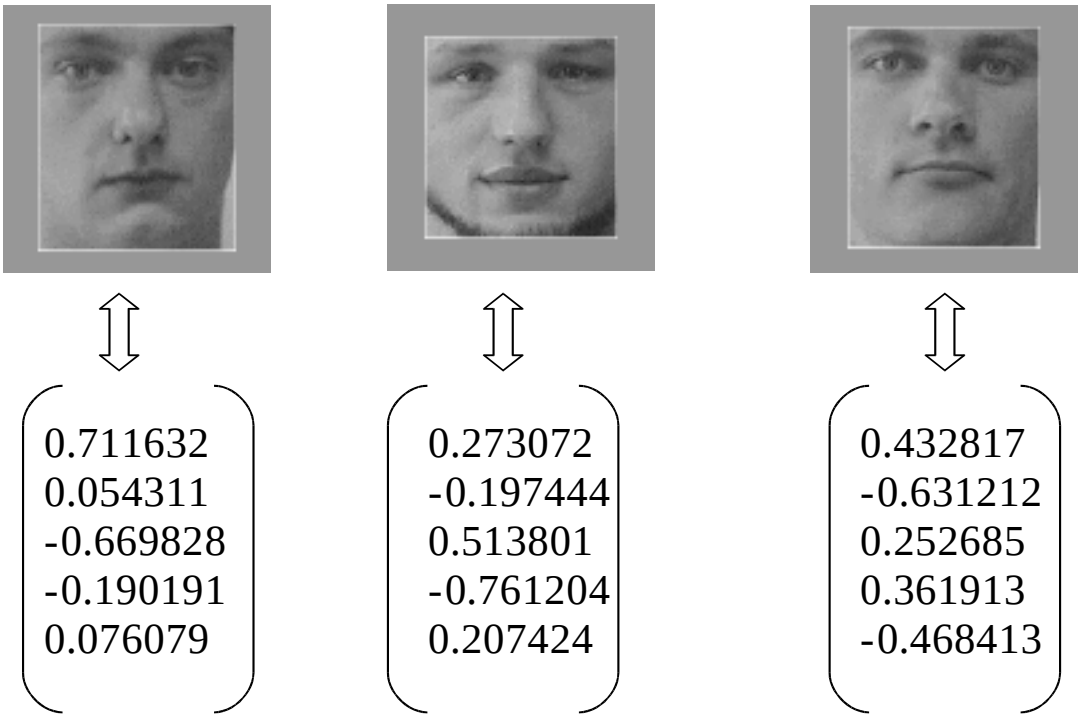
```
octave:6> v=cov(y)
v =

  730.18  461.69  534.37  570.64  507.03
  461.69  739.02  457.75  449.85  425.81
  534.37  457.75  674.33  528.90  488.97
  570.64  449.85  528.90  804.26  598.13
  507.03  425.81  488.97  598.13  917.86
```

Finalement, on peut appliquer la méthode eig sur la matrice carrée de covariance de taille nbfoto^2.

```
octave:7> [b,l]=eig(v)
b =

   0.711632   0.273072   0.432817   0.171945   0.449573
   0.054311  -0.197444  -0.631212   0.632283   0.399820
  -0.669828   0.513801   0.252685   0.199108   0.428772
  -0.190191  -0.761204   0.361913  -0.161553   0.476777
   0.076079   0.207424  -0.468413  -0.710570   0.476305

l =

   164.13221     0.00000     0.00000     0.00000     0.00000
     0.00000   196.24860     0.00000     0.00000     0.00000
     0.00000     0.00000   297.24862     0.00000     0.00000
     0.00000     0.00000     0.00000   415.25215     0.00000
     0.00000     0.00000     0.00000     0.00000  2792.76171
```


Note sur la méthode eig de Octave : Les eigenvalues (et eigenvecteurs) d'une matrice sont calculées à partir d'un processus commençant par une décomposition de Hessenberg suivi par une ou plusieurs décomposition de Schur.

Ci-dessus, la matrice carrée b contient les eigenvecteurs de chaque visage. Dans l'ordre de chargement des images, on a en partie :

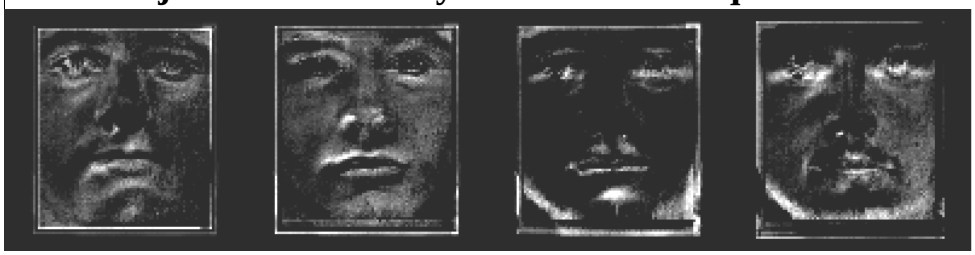


Note: la banque de données doit représenter le plus fidèlement possible les caractéristique d'un visage humain. Le nombre de eigenvalues dans les vecteurs est égal au nombre de visage introduit dans la banque pour l'échantillon.

Ces vecteurs m'intéresse que dans la mesure où ils vont me permettre de générer un espace. Dans celui-ci, chaque dimension sera un visage dont on aura tiré les caractères biométriques commun avec les autres visage, et sur lequel on les aura appliqués. Il ne reste plus qu'a décrire ces visages dans le plan, c'est-à-dire :

```
octave:8> t=y*b ;
octave:9> size (t)
ans =

  10000      5
octave:10> imshow (reshape(t,100,500))
```

On obtient dans 't' des visages synthétiques ou projetés : les eigenfaces. La dernière commande affiche ces visages après conversion dans un format ('dim (1)','dim(2)*nbfoto').



L'ensemble de ces eigenfaces sont en quelque sorte des clés publiques, diffusables. Je voudrais qu'elle soit exportée vers le système de personnalisation et vers le système d'authentification. Il est question de clé, de signatures, qui est quoi, qu'exporter ?

Considérations sur l'identification et authentification dans le projet.

Admettons que le moyen d'authentification est une signature et un moyen d'identifier, une clé. L'exemple de la transaction bancaire au GAB (Guichet Automatique de Billets) tend à confirmer cela : le porteur de la carte qui veut retirer de l'argent doit donner son code secret ou sa clé privée. Le terminal en lui-même, ne vérifie que l'identité du code secret tapé avec celui stocké dans la carte. (Hélas !) Il ne peut savoir (dans le temps imparti par la transaction) s'il le porteur de cette carte est véritablement la personne qui a ouvert le compte à la banque, « le porteur authentique ».

Le terminal proposé dans ce projet tient compte de cette problématique car ici, la clé secrète c'est le visage... Ou plus exactement les eigenvecteurs de ce visage : on est sûr que la personne qui retire l'argent est au moins présente devant le terminal qui prend sa photo (authentification). Cette photo depuis laquelle on calcule la clé privée doit être égale (« à peu prés ») à celle au préalablement calculée contenue dans la carte personnalisé (identité). En définitive, ces eigenvecteurs identifient et dans des conditions définies authentifient aussi, alors ils sont des clés-signatures.

Comme convenu, je vais considérer que le terminal certifieur à expédié cette clé au terminal de personnalisation par réseau sécurisé. Pour simuler cela, je sauvegarderai de mon environnement Octave uniquement 't' c'est-à-dire je détruirai tout sauf la variable 't'.

```
octave:11> who -variables
*** local user variables:

depth   dim     mmm     nbfoto  y    t

octave:12> clear depth ;clear dim
octave:12> clear mmm ; clear nbfoto
octave:12> clear y ;
octave:12> who –variables
*** local user variables:
t
```

En annexe, vous retrouverez les listings des codes fr_load.m et fr_databank.m qui constitue le programme du terminal certifieur.

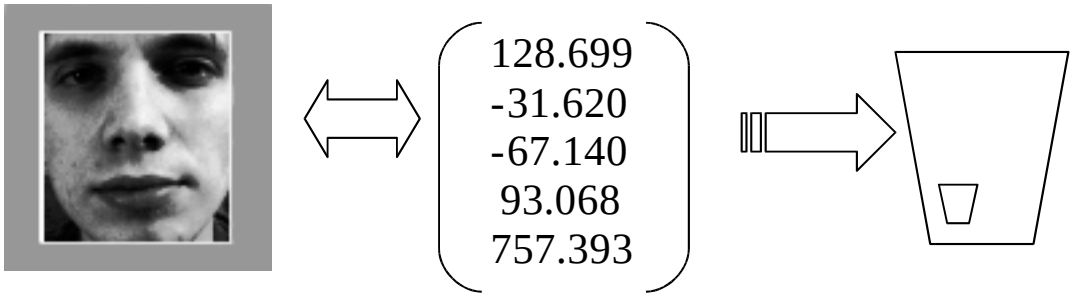
Projet industriel : système biométrique de reconnaissance

Implémentation du terminal de personnalisation

Le terminal de personnalisation a pour rôle d'enregistrer un nouvel utilisateur du système. Il prend une photo de celui-ci auquel on applique les même méthodes d'acquisition et de traitements d'image vues auparavant. Puis, il génère une signature clé privée à partir de la signature clé publique envoyé par le terminal certifieur. La signature privée sera enregistrée sur carte à puce. Ce terminal pourrait (devrait ?) également informer le terminal certifieur de l'abonnement d'un nouvel usager (stockage en base de données).

Le code de cette procédure est relativement restreint et rapide au vue de la procédure de terminal certifieur.

```
octave:13> load -force 1mat.pgm.oct
octave:14> who -variables
*** local user variables:

ncol   dims    nbfoto  X   t

octave:15> yprime=reshape(X,dims(1)*dims(2),1);
octave:16> yprimec=yprime-ones(dims(1)*dims(2),1)*ncol/2;
octave:17> vprime=cov(t,yprimec);
octave:18> vprime =
                        128.699
                        -31.620
                        -67.140
                         93.068
                        757.393
octave:19> nbphoto
nbfoto = 5
```



Notes sur le stockage en carte : l'intérêt est de stocker cette signature sur une carte à puce limitée en taille mémoire. On constate que le nombre de eigenvalues de cette signature correspond également au nombre de photos initiales de la banque, d'où l'importance majeure de la constitution d'un échantillon mesuré de qualité.

En annexe, vous retrouverez le listing du code de fr_load.m et de fr_perso.m

Implémentation du terminal d'authentification

Le terminal d'authentification est le terminal usuel sur lequel l'usager du système viendra de faire identifier et authentifier afin par d'exemple d'acheter un service ou produit qui intégrerai ce système. Le terminal opère comme le terminal de personnalisation, il génère une autre signature privée avec la photo qu'il prend de l'usager. A l'insertion de la carte, il lit la signature de la carte, prend le cliché. Il demande la clé publique au terminal certifieur et compare la signature authentique (sur la carte) avec la signature nouvellement générée. Et, selon qu'elle est authentique ou non, il débite le compte de l'usager ou l'informe du refus.

Le code est similaire au code du terminal de personnalisation, il contient la procédure d'authentification en plus. Au plus simple, elle consiste en un rapport : on rapporte la clé privée authentique à la clé privée nouvellement calculée et suspecte. La photo authentique (clé privée authentique : vprime) n'est pas similiare à la photo suspecte (clé privée suspecte : vseconde) alors ce rapport indique des taux d'identité des eigenvalues (r).

```
octave:20> r=vseconde./vprime
r =
   0.75486
   0.29523
   0.39279
  -0.29557
   1.68597
octave:21> seuil=[0.00;0.20;0.40;0.60;1.00] ;
octave:22> r>seuil
ans =
   1
   1
   0          3 caractères biométriques des 5 sont satisfaits.
   0
   1
```

La première eigenvalue (0.75486 =~ 75%) se rapporte au *visage synthétique mineur* celui auquel il y a le moins de caractères biométriques communs dans les visages de la banque de données. La dernière eigenvalue (1.68597 =~ 168%) se rapporte au *visage synthétique majeur* celui auquel il y a le plus de caratères biométriques communs dans les visages de la banque de données. La méthode du décisionnel par seuillage ainsi que les valeurs de seuil sont choisis arbitrairement (pour leur simplicité) dans cet exemple mais reflètent tout de même la considération de l'importance croissante des eigenvalues dans l'échantillon.

Elles pourraient (devraient ?) être déduites d'un protocole. Le but de celui-ci serait de dégager un seuillage moyen sur plusieurs autres systèmes munis du même échantillon, avec une même photo authentique. Puis chaque terminal traiterait avec plusieurs photos authentiques (et donc des clés privées suspectes) pour simuler différentes poses, différents éclairages d'un même personnage.

En annexe, vous retrouverez le listing du code de fr_load.m, de fr_auth.m ainsi que le script de démonstration qui décrit comment utiliser tous ces scripts des terminaux.

7. Bilan et perspectives d'évolution dans le domaine

J'ai réussi à mettre en place un prototype qui fonctionne grâce à ue implémentation « allégée » (simplissime !). Alors évidemment, cette implémentation a écarté de nombreux aspects, que je voulais tout de même traiter. Tous les points abordés ci-dessous sont des facteursde risques à la conception, lors des développements et de l'exploitation d'un système de ce type (appelé 3-tiers) à plus grande échelle, à un stade industriel.

Comment améliorer l'acquisition des photos ? Notamment comment réduire l'impact de l'illumination sur la reconnaissance ?

L'impact de l'illuminationsur la reconnaissance est un risque révélé dès l'étude de l'art. En l'état, c'est surtout le dispositif normatif qui assure un environnement maîtrisé en dictant des régles de lumière, de dispostion... disons d'acquisition. Mais que faire si l'on veut par exmple réduire l'encombrement de ce dispositif sans rogner sur la qualité de la reconnaissance ? Soit : Que faire si l'on veut modifier l'ajustement (tel que décrit auparavant dans *« considérations sur la complexité des traitements »*) sans détériorer l'efficacité de l'authentification ?

Il existe plusieurs solutions pour remplacer, complémenter, affiner la méthode de détection des reliefs. En l'état, il s'agit simplement d'une fonction de covariance pixel à pixel (cov).

De meilleurs solutions ordonnées dans l'ordre croissant de leur complexité (cf . Etat de l'art) :

- Coefficient de corrélation (corrcoef)

Fonction disponible dans Octave, utilisable telle quelle sans modification significative au regard de la compléxité de traitement. Nécessairement plus lent en temps d'exécution que la covariance.

- FisherFace (Fisher Linear Discriminant)

Cette méthode remplace tout ou partiee du code existant des terminaux. C'est acutellement la technique la plus poussée de correction lumineuse statistique se basant sur la méthode (eigenface) qui génère les eigenvalues.

- Principal Component Analysis (PCA)

A l'inverse des deux méthodes de correction « en aval », celle-ci résout le problème « en amount », dès l'acquisition, ce gain de temps correspond à un gain en temps de calcul en faisant acquisition des parties « utiles », fortement discriminantes, de l'images (en fait l'image à traiter devient plus petite)

- Independant Component Analysis (ICA)

Dérivé du PCA, lui aussi traite le problème « en amount ». J'ai n'ai pas pu receuillir un faisceau d'informations suffisemment cohérent.

Vers une industrialisation du procédé...

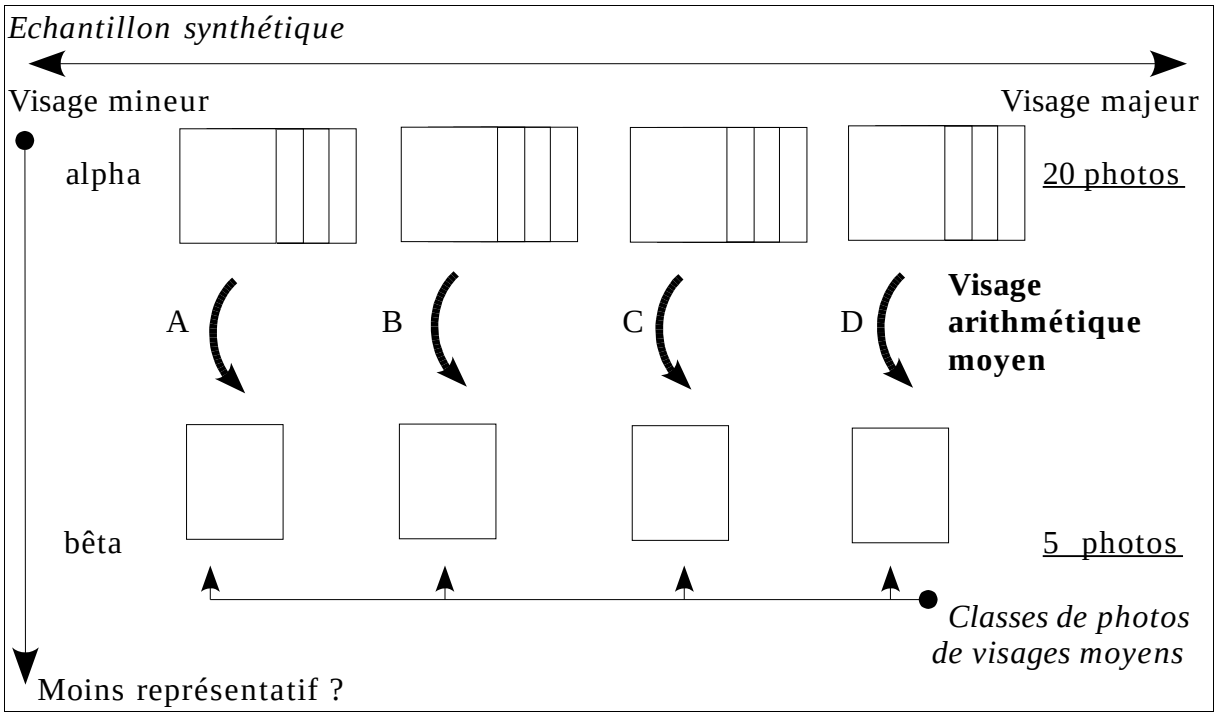
En raison des moyens du projet, l'échantillon et le jeu de test sont, il est vrai, (tout à fait) restrincts. Une question légitime est donc : comment ce système se comporterait une fois mise à l'echelle insdustrielle ?

Il faudrait agrandir l'échantillon, les travaux de référence en la matière recommendent un échantillon de visages composites de la taille de la surface d'une image à reconnaître. Donc, dans cette étude une image qui fait 100 pixels sur 100 pixels pourra être reconnu optimalement grâce à un échantillon de 10000 visages ! Clairement mon échantillon de six visages est *nettement* trop réduit. Néanmoins, si celui-ci respecte déjà quatre des cinqs critères qui définissent un bon échantillon (durable, homogène, varié représentatif, manque donc taille adaptée), cela provoque déjà des résultats médiocres vis à vis du décisionnel par seuillage : mon micro-échantillon expérimental est à la mesure du nombre de visage que j'ai testé, qu'il serve à mettre en évidence justemment ce besoin vital pour l'industrialisation !

Considérations sur le stokage de l'information...

Cette voie vers l'industrialisation demande donc un échantillon de 10000 visages synthétiques. La clé-signature publique comptera alors 10000 eigenvecteurs, ainsi que chaque clé-signature privée. Cela n'a plus rien à voir avec les clés-signatures générées et testées dans le protocole décrit dans ce document alors, le temps d'exécution aussi. Ce temps que l'on pourrait penser devenu trop long, ne pose pas de nouveau problème : lors de la personnalisation il survient une et unique fois, lors de la création d'un usager; cependant est bien pire, ce même temps est multiplié par le nombre de fois qui usager faudra s'authentifier. Je n'ai pas précisément estimer ce temps mais au regard de la puissance de calcul d'une carte à puce et de la taille de ce type de clé, le temps de réponse résultant ne serait pas probablement acceptable. Ce dernier pour les automates doit être de quelques secondes au maximun.

Comment réduire la taille d'un échantillon sans pour autant perdre de l'information utile ? Ou encore comment compresser un échantillon sans perdre un seul caratère biométrique d'un visage synthétique, et cela même s'il s'agit d'un appartenant au visage mineur ? Il est tout à fait envisageable de créer des classes de visages synthétiques dits secondaires englobant plusieurs visages synthétiques (primaires). En revanche, il est légitime de se demander si une telle opération n'altérera pas la qualité des critères de l'échantillon, notamment la représentativité.



Compression de l'échantillon au prix de la représentativité ?

Vers un système intègre et plus sécurisé...

Tout au long de ce document, le terme clé ou signature s'est trouvé lentement mais surement remplaçé par le terme agrégé clé-signature. Cette volonté de précision lexicale renvoie à la vonlonté de créer un système d'identification et d'authentification par analogie aux systèmes cryptographiques asymétriques. Est-ce que je n'aurais pas (seulement) opérer une sorte de glissement sémantique ? Pas du tout. Cependant, peut-être n'avez vous pas remarqué que tous les terminaux se partagent la même et unique clé-signature publique (CSP) ? (cf. Diagramme de fonctionnnement des terminaux)

Ceci n'est pas le cas des systèmes experts qui m'ont servi de « modèles », évidemment cela revient à dire, qu'en l'état, il y a un « trou de sécurité » (plutôt même un gouffre). Cela perdurerait, si l'architecte d'un tel système oubliait d'introduire la notion de défi, la notion de certificat, en implémentant un aléa. Cet aléa appliqué sur la CSP initiale en fournirait une seconde pour un terminal de personnalisation donné, et pour autant de terminaux d'authentification, ceci pour autant que l'administrateur du terminal de personnalisation l'autorise. En outre, l'aléa serait retourné au terminal certifieur, qui recalculerait la CSP seconde du terminal de personnalisation. Aussi, le terminal certifieur, toujours muni de la CSP (initiale) ainsi que de l'aléa, pourrait émettre un certificat au terminal de personnalisation.

Son utilité réside dans l'assurance de l'identité et de la confiance que donne ce tiers (dit de confiance) à l'autre partie. Il permet aussi de se faire réidentifier (et émettre un nouveau certificat) par ce tiers, cas de perte ou de piratage de la CSP seconde au niveau du terminal de personnalisatoin afin qu'il se fasse renvyer une CSP première.

A noter, qu'un pirate qui s'emparerait de la CSP devrait s'emparer aussi de l'aléa pour n'obtenir que la CSP seconde d'un terminal de personnalisation donné. Au mieux et probablement pour pas très longtemps, puisque cette CSP seconde peut-être recalculée « à volonté » avec regénération d'un aléa du terminal de personnalisation. De plus ce dernier pourrait sans peine remarquer le larin dès la première utilisation, car seul lui-même et ces terminaux authentifications associés (qui sont donc connus de lui) ont normalement accès à une CSP seconde !

Une remarque sur ce système « version seconde » en définitive, concerne l'interopératibilité des terminaux d'authentification qui ne sont pas associés avec des terminaux de personnalisation. Cette fonctionnalité (à l'instar les retraits bancaires faits sur des GAB d'autres banques que celle où l'on est client) complique à peine un peu le modèle évoqué ci-dessus. En effet, il « suffit » d'implanter une gestion de certificat seconde au niveau de la chaine personnalisation-authentification (comme l'est la permière au niveau de la chaine certification-personnalisation).

En annexe, vous retrouverez le modèle mathématique de ce système « version seconde».

Plus loin, autres facteurs de risques : la fraude biologique n'est plus une fiction...

Il existe bien des façons de tromper un système informatique de reconnaissance, l'usurpation d'identité y tient une place prépondérante. Les récentes avancées dans le domaine de la recherche biologique et des techniques de greffes ouvrent la voie vers des fraudes biologiques dont l'attaque agit directement sur l'objet de la reconnaissance : le vivant.

Il est dorénavant possible de greffer un visage humain sur une personne ! Mais ces techniques de greffe ne sont pas nées d'hier et la greffe de peau avait déjà mis en défaut les systèmes de reconnaissance biométrique d'empreinte digitale. Cette attaque déjà lourde et complexe à réaliser consiste à se faire greffer de la peau des doigts d'une personne autorisée. Mais les systèmes de reconnaissance faciale semblait être pas encore concernés. Ce n'est désormais plus le cas.

Toutefois, peut-être sans le savoir, ce système propose déjà une solution à cette problèmatique, solution qui emprunte aux travaux de recherche sur la reconnaissance d'expression du visage. Il semble concevable d'adapter le système décrit dans ce document dans cette optique : reconnaître un visage faisant une mimique précise. Et si l'utilisateur en phase de personnalisation produisait une mimique, connue de lui seul, afin qu'elle génère une clé-signature privée spécifique ? Evidemment, il serait nécessaire de modifié le critère normale de neutralité de l'expression du visage et de mesurer son impact sur la reconnaissance finale de l'authentification.

Si ce système peut être efficace, alors une greffe de visage sans connaître la mimique de reconnaissance serait inutile. Et même la connaissant, il faudrait encore la reproduire avec autant d'habilité que son propriétaire, lequel manque pourrait être dectecté.

CONCLUSION

Les systèmes bimétriques connaissent des difficultés qui ne sont plus vraiment dues à la maîtrise de l'environnement dans lesquels ils fonctionnent. Celle-ci est en constant accroissement. Un véritable problème réside plutôt dans la façon dont ils sont perçus : intrusifs, complexes, non-fiables voire inutiles. Tout au long de cette étude, j'ai tenu à faire en sorte que :

les données privées ne soient pas diffusable ni diffusées, ni même stockées sans consentement et que les utilisateurs aient l'assurance que c'est le système qui s'adapte aux spécificités humaines et non l'inverse;

l'architecture, les traitements, les résultats soient d'une simplicité de compréhension à la mesure des exigences des fonctionnalités requises;

l'efficacité à la mesure de la confiance accordée au système grandisse par l'étude et la recherche de solutions issue de références reconnues;

l'intérêt, au niveau des utilisateurs, des développeurs, des industriels, soit au centre de cette étude en considération des problèmatiques de chacun.

Parce que ces systèmes peuvent offrir à chacun une transparence, une aisance, une efficacité, et une plus-value dont il serait dommage de se priver.

REFERENCES

Turk, M., and Pentland, A., "Eigenfaces for Recognition", Journal of Neuroscience, vol.3, no1 1991

Belhumeur, P., Hespanha, J., and Kriekman, D., "Eigenfaces vs. Fisherfaces : Recognition using class specific linear projection", IEEE Trans. Pattern Analysis Machine Intelligence, vol.19 ,no.7, pp.711-720, July 1997

Pentland , A., Moghaddam, B., Starner, T., and Turk, M., "View-based and Modular Eigenspaces for Face Recognition" , proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, pp.84-91, Seattle, WA,1994

Meng Joo E*r, Member IEEE*, Shiqian W*u, Member IEEE*, Juwei Lu*, Student Member IEEE*, and Hock Lye Toh*, Member IEEE. "Face Recognition With Radial Basis Function (RBF) Neural Networks"*

Middle East Technical University, Department of Computer Engineering, Senior Design Project and Seminar, *"Face Recognition using EigenFaces"*

A. Belaïd, Y. Belaïd, "Reconnaissance des formes : Méthodeset applications", InterEditions, janvier 1992.

http://www-rocq.inria.fr
http://www.lasmea.univ-bpclermont.fr
http://www.cim.mcgill.ca
http://www.csdl.computer.org
http://www.ieee.org
http://www.mrc-cbu.cam.ac.uk
http://www.orensic.shef.ac.uk
http://www.vision.ai.uiuc.edu
http://www.open.brain.riken.go.jp
http://www.sources.redhat.com/gsl
http://www.r-project.org
http://www.octave.org
http://www.mathworks.fr
http://www.gnu.org
http://www.fsf.org

ANNEXES

L'état de l'art

Projet industriel : système biométrique de reconnaissance

Introduction : Motivation et Intérêt du projet ...

L'utilisation d'applications d'identification et d'authenfication est de plus en plus répandue. Ce fait reflète le besoin pour le système d'information, de reconnaître un usager qui va par exemple, l'employer à une tâche ou service déterminé.

La méthode d'authentification la plus répandue est celle du code secret, un numéro à retenir et à restituer par exemple lors d'un retrait à une billetterie automatique ; le terminal doit s'assurer que le porteur de la carte est bien son propriétaire. Plusieurs inconvénients existent :

- le temps de saisi relativement long au regard de la vitesse de traitement du terminal ;
- que faire si l'on oublie son code (multiplication des codes secrets à retenir) ?
- que faire si un propriétaire de carte se fait voler son code ?

Il faudrait concevoir un système « naturel » non-intrusif qui permette à la personne qui se fait authentifier, identifier d'être le plus passif possible.

Description du projet

L'image du visage a authentifier sera stockée, dans un premier temps, dans la mémoire de la carte à puce. Le terminal devra accorder l'accès lorsqu'il comparera l'image testée à la référence stockée dans sa base de données.

Un peu d'histoire…

De 1994 à 1997, un programme de recherche sous l'égide du DARPA et du laboratoire de recherche de l'armée des U.S.A est chargé d'analyser la performance d'algorithme de reconnaissance de visage. Les résultats des tests de ce programme ont été salués unanimement par la communauté de chercheurs en ce domaine. Une base de données de visage, ainsi que ces résultats sont toujours « disponible ».

Dès lors cela a inspiré diverses approches. Actuellement, principalement les universitaires et ingénieurs chinois et américains ont développé leur approche.

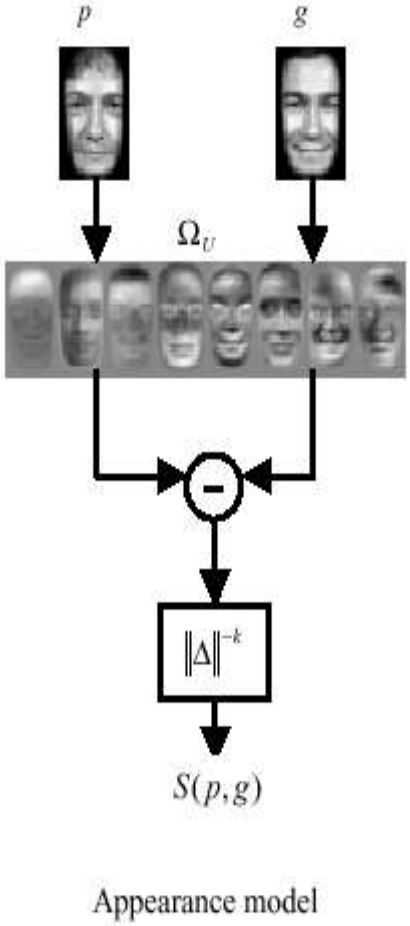
Mitsubishi mène un projet de recherche appelé MERL dans le domaine du computer vision, consistant à donner à la machine « notre faculté de voir » : une *approche neuronale* et *eigenface*. (Tsinghua University, Beijing). Les universitaires et ingénieurs américains travaillent davantage sur une *approche par modéle et eigenface* (MIT, Western Cafornia University, Stanford University*).*

Etat de l'art : approches et modèles existants...

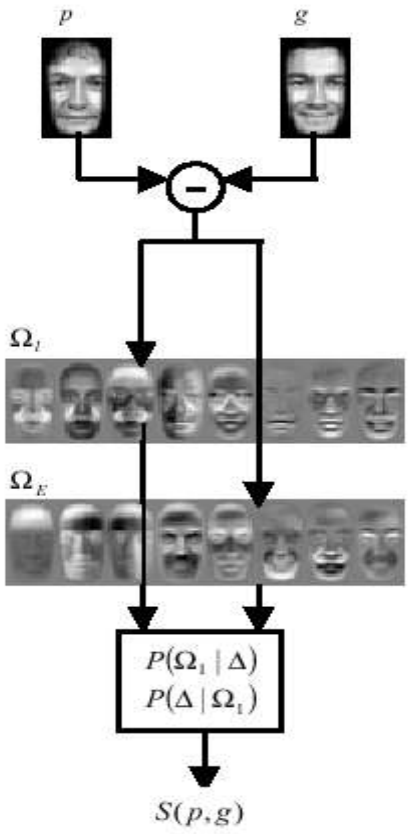
Les méthodes de reconnaissance étudiées sont les suivantes :

EigenFace (Eigenface method), correspondance de modèle (template matching), correspondance graphique (graph matching), sous espace linéaire (linear subspace method), analyse neuronale (neural network method) et la méthode fisherface.

- L'approche *eigenface* applique la transformation de Karhonen-Loeve l'extraction de données (KLE). Cela réduit grandement les tailles d'images et maintient un temps de calcul raisonnable et une discrimination efficace. Une approche d'origine anthropologique inventée en Allemagne très prisée. La méthode *Fisherface*, et un algorithme encore plus élaboré que *eigenface*. Il emploie le discriminant linéaire de Fischer (FLD).



Appearance model



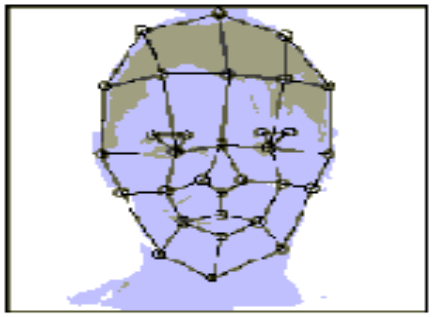
Discriminative model

- L'approche par *réseau neuronaux*, apporte une solution de pointe. Elle réalise des modèles sophistiqués d'espérance selon les densité dans les formes lors de la phase de reconnaissance. [JWLT02]. La Chine travaille beaucoup sur cette approche.

- La *correspondance de modèle* opère une comparaison directe de segment d'image. Efficace seulement si les images-test ont la même échelle, orientation, illumination. Méthode résolument la moins complexe mais un temps de pré traitement de l'image est nécessaire. Cette méthode a été poussée avec une annexe la PCA (Principal Component Analysis).

- Dans *correspondance graphique*, des graphs représentent des objets sectorisés par segments et nommées selon des paramètres tels la distance géométrique, la couleurs, la luminosité , etc.. . Une méthode répandue, qui a déjà beaucoup servie notamment dans la reconnaissance des expressions du visage (colére, joie, neutre, etc..).

1.définition ou analyse des bords de segments (vertices)
2.modélisation informatique
3.reconnaissance grâce à la capacité du modéle à ''distendre'' ses segments comparée à une image de reférence stockée.



- Dans la méthode du *sous espace linéaire*, plusieurs images (typiquement 3 images) du visage objet sont prises dans des conditions de luminosité différentes. Un espace 3D est donc construit, la reconnaissance s'opère entre l'image à tester et chaque sous-espace linéaire. Puis, par calcul de la distance entre chacun de ces sous-espaces que l'on veut le plus faible, on détermine l'authentification.

Choix d'une approche: les avantages et les inconvénients

Projet industriel : système biométrique de reconnaissance

Après délibérations en présence du maître d'oeuvre du projet M. Baloche et de M. Signolle, nous avons décidé de choisir une approche basée sur la reconnaissance d'apparence et en particulier l'approche Eigenface pour plusieurs raisons :

- le traitement Eigenface utilise directement des données brutes issue directement d'un processus d'acquisition simple. Les images en niveaux de gris sont l'objet d'une reconnaissance qui ne nécessite pas de coûteux traitements de bas niveaux (tels conversions, formatages, etc..).

- cette approche ne requière pas d'expertise géométrique dans les traitements des reflets lumineux tels des dispersions, réractions sur les visages.

- une « compression » des données est intégrée dans la méthode, le résultat est un sous espace représentatif de l'espace père (les images).

- La reconnaissance est relativement rapide et efficace en rapport à d'autres approches connues, notamment celles par correspondance (template matching).

Cette approche connaît aussi des limitations, et dans le développement du projet, il faudra y prêter une attention particulière afin de ne pas nuire à la qualité de l'implémentation.

- cette méthode bas »e sur la reconnaissance de caratères codés par des teintes de gris est sensible à la pose et à l'illumination de l'objet à reconnaître. L'approche par projection FisherFace résout ces problèmes en maximisant la différence entre différentes classes de teintes. Cependant, trouver un point optimal théoriquement avec cette approche est réputé comme étant infaisable.

- cette méthode est très sensible au facteur d'echelle et de zoom lors de l'acquisition de l'image. Il faudra robableme t prévoir un sous-traitement d'acquisition normalisé. A noter que la méthode PCA (Principal Component Analysis) offre une solution efficace d'acquisition de portion hautement significative dans l'image.

Quelques mots en forme de parenthéses sur le choix PCA :

PCA doit faire surgir dans une scène l'objet à reconnaître ou sur l'objet à reconnaître fait surgir des portions qui apparaissent les plus hautement significatives. Bien qu'il s'agisse d'un traitement itératif (calcul au $n^{éme}$ ordre) il est réputé très rapide. Une méthode composite appelée Kernel PCA sert en imagerie médicale à la classification (GREYC).

Projet industriel : système biométrique de reconnaissance

Durant mes investiations, j'ai trouvé des travau de recherches très intéressant utilisant cette méthode mais malheureusement, ces sources sont rares, expliquent globalement ou théoriquement et souvent ne presentent que des résultats sans expliquer le foncitonnement de la méthode, que je n'ai eu que le temps de deviner. Il m'aurait fallu nettement plus de temps pour expérimenter et mettre en oeuvre une telle solution. Pour ces raisons cette option dans le projet qui a fait l'objet de nombreuses discussion avec M. Baloche, n'a pas été retenue. Elle mériterait cependant d'être étudiée de plus près...

- cette méthode requière la constitution d'une banque d'images, un échantillon de l'objet à reconnaître, base indispensable pour opérer alors une comparaison sur un sous-ensemble représentatif de cet objet. Cela contrarie fortement une mise à jour fréquente de cette banque. Ceci dit cette mise à jour à assez peu d'intérêt si cette banque est bien constituée.

- la projection d'un objet donnée dans l'espace de cet échantillon demande des calculs pixel à pixel de ces images, ce qui revient au mieux à une reconnaissance par une banque de modèle (model template matching). En revanche, son intérêt reste supérieur dans les cas où la taille de cet échantillon dépasse les dimensions de l'image de l'objet donné.

Les scripts des terminaux

Le script de préparation des données

```
#!/bin/bash
#   Script qui prepare l' image .pgm donnée
#  en paramètre en script de donnée initiale octave
#  .pgm.oct. Le commutateur -rm autorise la destruction
#  du .pgm original
#
#   Renvoie erreur 1 : aucun paramètre
#                  2 : file-to-convert n'est pas un fichier regulier
#
#   Auteur   : Mathieu Deschamps
#              mathdesc@scourge.fr
#   Date     : 11/02/2004
#   IUT de Caen -- Licence Pro. M.C.A
if [ -z "$1" ]
then
  echo "Usage: `basename $0` filename-to-convert {-rm}"
  exit 1
fi

if [ ! -f "$1" ]
then
  echo "Error: Unable to find regular file $1"
  exit 2
fi

NEWFILENAME="$1.oct"
FILENAME="$1"

echo ".PGM File : $FILENAME"
echo ".PGM.OCT File : $NEWFILENAME"


REMOVE=""

if [ "$2" = "-rm" ]
then
  REMOVE="delete"
fi

NBLINES=`wc -l $FILENAME | cut -f1 -d' '`
X=`head -n4 $FILENAME | tail -n2 | head -n1 | cut -f1 -d' ' `
Y=`head -n4 $FILENAME | tail -n2 | head -n1 | cut -f2 -d' ' `
NCOL=`head -n4 $FILENAME | tail -n2 | tail -n1`

echo "#P3" >$NEWFILENAME
echo "#This file has been generated by $0 for Octave load function"  >>$NEWFILENAME

echo "# name: dims"  >>$NEWFILENAME
echo "# type: matrix"  >>$NEWFILENAME
echo "# rows: 1"  >>$NEWFILENAME
echo "# columns: 2"  >>$NEWFILENAME
echo "$X $Y" >>$NEWFILENAME

echo "# name: ncol" >>$NEWFILENAME
echo "# type: scalar" >>$NEWFILENAME
echo "$NCOL" >>$NEWFILENAME

echo "# name: X"  >>$NEWFILENAME
echo "# type: matrix"  >>$NEWFILENAME
echo "# rows: $X"  >>$NEWFILENAME
echo "# columns: $Y"  >>$NEWFILENAME

rm -f temp
touch temp
vi "$FILENAME" <<!
4dd
:w! temp
:q!
!
echo "Header: 14 lines written."
echo "Image : $NBLINES lines written."

cat temp  >>$NEWFILENAME
rm -f temp

if [ -n "$REMOVE"  ]
then
    rm -f $FILENAME
        echo "Original file $1 has been removed."
fi
exit 0
```

Projet industriel : système biométrique de reconnaissance

Le script de traitement des photos

```
#!/bin/bash
#
# traitphoto.sh est un script qui appelle
#  prepare.sh sur toute les photos du repertoire
#  dans le repertoire courant. Il genere egalement
#  un fichier err_prep.log
#
# Renvoie  1 : aucun paramètre
#
#
#  Auteur       : Mathieu Deschamps
#           mathdesc@yahoo.fr
#  Date   : 11/02/2004
#  IUT de Caen -- Licence Pro. M.C.A
#
#

REMOVE=""

if [ -n "$1" ]
then
   if [ "$1" = "-rm" ]
   then
          REMOVE="delete"
   else
          echo "Usage: `basename $0` {-rm}"
          echo " {-rm} lets the original .pgm file be deleted"
          exit 1
   fi
fi



echo "#   Erreurs générées par prepare.sh" >err_prep.log
echo "#  lors de la préparation des données" >>err_prep.log
echo ">>Begin data conversion"

i=1
file="NIX rocks!"

while [ -n "$file" ]
do
 file="`/bin/ls -m *.pgm | cut -f"$i" -d',' -s`"
 echo "$i° file : $file"
   if [  -n "$file" ]
   then
          if [ -n "$REMOVE"  ]
            then
               sh prepare.sh $file -rm 2>>err_prep.log
                   else
              sh prepare.sh $file 2>>err_prep.log
          fi
   fi
 i=`expr $i + 1`
done

echo " No more file to trait... "
echo ">>End of data conversion"
exit 0
```


Projet industriel : système biométrique de reconnaissance

Acquisition d'une image : FR_LOAD.M

```
function [y,depth,dim,mmm,nbfoto] = fr_load (C_Taille)
%
% Fr_load admet la taille X*Y de l'image en paramètre.
%  Elle recherche tout les fichier .pgm.oct du répertoire
%  courant. Elle met en forme toutes les images au format
%  vecteur colonne. Elle vérifie également la cohérence
%  des données de ces images.
%
%         nbfoto : nombre de photo du répertoire courant
%         y : Banque de données vecteur colonne image de taille
%                [C_Taille,nbfoto]
%         depth : profondeur de couleur des images chargées
%         dim    : dimension [X,Y] d'une image
%         mmm : vecteur contenant des données calculée sur y
%                [Teinte min, max , moyenne]
%         C_Taille : ceci est une contrainte (repr X*Y)
%
%
%       Auteur  : Mathieu Deschamps
%                 mathdesc@scourge.fr
%       Date    : 11/02/2004
%       IUT de Caen -- Licence Pro. M.C.A
%


   f=popen ("ls -1 *.pgm.oct | wc -l","r");
   n=fscanf(f,"%d");nbfoto=n;
   printf (" %d images .pgm.oct trouvées \n",n);fclose(f);
   f=popen("ls -1 *.pgm.oct","r");

   for i=1:n
          file=fscanf(f,"%s",1);
          cmd=sprintf("load -force %s",file);file
          eval(cmd);

           if (dims(1)*dims(2)!=C_Taille)
                 printf("%s: La taille de l'image [%i,%i] et le paramétre \nde
la fonction %i doivent correspondre !\n",file,dims(1),dims(2),C_Taille);
                 clear;
          end

          X=reshape(X,C_Taille,1);
          depth=ncol;
          dim=dims;
                 if !exist("y")
                        y=X;
                 else
                        y=[y,X];
                 end
   end
   mmm=[min(min(y)),max(max(y)),mean(mean(y))];
   if (  ((mmm(1)<0) || (mmm(1)>255))  ||  ((mmm(2)<0) || (mmm(2)>255)) || ((mmm
(2)<0) || (mmm(2)>255))   )
   printf("min: %i , max: %i , moyenne: %i valeur hors norme ! ",mmm(1),mmm(2),
mmm(3));
   clear;
   end

   s=size(y);

   if (s(1)!=C_Taille || s(2)!=n)
   printf ("Erreur lors du chargement ...");clear;
   end

   fclose (f);

save load.save
```


Constitution d'une banque d'image : FR_DATABANK.M

```
        function t = fr_databank (depth,dim,nbfoto,y,save,verbose,draw)
        %
        %  Fr_databank  constitue  la  banque  de  donnée.  Cette  fonction
effectue
        %   un traitement d'image puis renvoie le vecteur clé publique t,
        %   c'est-à-dire les EigenFaces.
        %        nbfoto : nombre de photo du répertoire courant
        %        y : Banque de données vecteur colonne image de taille
        %              [C_Taille,nbfoto]
        %        depth : profondeur de couleur des images chargées
        %        dim   : dimension [X,Y] d'une image
        %              [Teinte min, max , moyenne]
        %
        %        DEBUG FLAGS
        %        save    : mis a 1 cela sauve de l'environnement
        %                  (dans le fichier databank.save)
        %        verbose  :  mis  a  1  cela  affiche  les  resultats
intermédiares
        %        draw  : mis a 1 cela affiche les EigenFace en fin
        %
        %
        %       Auteur  : Mathieu Deschamps
        %                 mathdesc@yahoo.fr
        %       Date    : 11/02/2004
        %       IUT de Caen -- Licence Pro. M.C.A
        %


        yc=y-ones(dim(1)*dim(2),nbfoto)*depth/2;
        if (verbose==1)
           v=cov(yc)
        else
           v=cov(yc);
        end
        if (verbose==1)
           [b,l]=eig(v)
        else
           [b,l]=eig(v);
        end
           t = yc*b;
        if (size(t)!=size(yc))
           printf("Erreur lors du traitement : taille incohérente");
        end
        if (save==1)
           save "databank.save";
        end
        printf("Banque de données EigenFace constituée\n");
        if (draw==1)
           imshow(reshape(t,dim(1),nbfoto*dim(2)));
        end
```


Personnalisation d'une carte : FR_PERSO.M

```
function vprime = fr_perso (file,t,C_Taille, save,verbose)
%
% Fr_perso 'abonne' un usager du systeme. Cette fonction crée une
%    cle privée-signature de cette abonnee depuis sa photo .
pgm.oct
%   et la clé publique t, le vecteur de EigenFaces.
%
%       file : nom du fichier pgm.oct à ajouter
%       t    : vecteur EigenFaces
%       C_Taille : ceci est une contrainte (repr X*Y)
%       vprime : Valeur renvoyee. Clé privée de l'usager.
%
%       DEBUG FLAGS
%       save    : mis a 1 cela sauve de l'environnement
%                 (dans le fichier perso.save)
%       verbose  :  mis  a  1  cela  affiche  les  resultats
intermédiares
%
%      Auteur  : Mathieu Deschamps
%                mathdesc@scourge.fr
%      Date    : 11/02/2004
%      IUT de Caen -- Licence Pro. M.C.A
%

  cmd=sprintf("load -force %s",file);file
eval(cmd)
if (dims(1)*dims(2)!=C_Taille)
                printf("%s: La taille de l'image [%i,%i] et le
paramétre \nde la fonction %i doivent correspondre !\n",file,dims(1),dims
(2),C_Taille);
end

yprime=reshape(X,dims(1)*dims(2),1);
yprimec=yprime-ones(dims(1)*dims(2),1)*ncol/2;


if (verbose==1)
 vprime=cov(t,yprimec)
else
 vprime=cov(t,yprimec);
end

if (save==1)
  save "perso.save";
end
printf ("Clé privée générée...\n");
```


Authentification et identification : FR_AUTH.M

```
        function    vseconde    =    fr_auth    (file,t,vprime,C_Taille,
save,verbose)
        %
        % Fr_auth crée un signature quand un usager est pris en photo
(file)
        %     et qu'il fourni sa clé privée vprime qu'il saisit (en
parmatère).
        %    cette singaute est donc fait grace a sa photo .pgm.oct, sa
clé privee
        %   et la clé publique t, le vecteur de EigenFaces.
        %
        %       file : nom du fichier pgm.oct à authentifier
        %       t    : vecteur EigenFaces
        %       vprime : vecteur clé privée à vérifier
        %       C_Taille : ceci est une contrainte (repr X*Y)
        %       vseconde : Valeur Renvoyée. La clé privée de l'usager
        %       (dans l'ideal doit etre egal à vprime)
        %
        %       DEBUG FLAGS
        %       save    : mis a 1 cela sauve de l'environnement
        %                 (dans le fichier auth.save)
        %       verbose   :   mis   a   1   cela   affiche   les   resultats
intermédiares
        %
        % Auteur      : Mathieu Deschamps
        %         mathdesc@scourge.fr
        % Date  : 11/02/2004
        % IUT de Caen -- Licence Pro. M.C.A
        %

        cmd=sprintf("load -force %s",file);file
        eval(cmd)
        if (dims(1)*dims(2)!=C_Taille)
                    printf("%s: La taille de l'image [%i,%i] et le
paramétre \nde la fonction %i doivent correspondre !\n",file,dims(1),dims
(2),C_Taille);
        end

        yseconde=reshape(X,dims(1)*dims(2),1);
        ysecondec=yseconde-ones(dims(1)*dims(2),1)*ncol/2;

        if (verbose==1)
         vseconde=cov(t,yseconde)
        else
         vseconde=cov(t,yseconde);
        end

        if (save==1)
           save "auth.save";
        end
        printf("Clé privée à authentifier générée...\n");
```

Exemple d'utilisation des fonctions : diary.txt


```
octave:2> [y,depth,dim,mmm,nbfoto]=fr_load(10000);
 5 images .pgm.oct trouvées
file = 1alban.pgm.oct
file = 1etienne.pgm.oct
file = 1jerome.pgm.oct
file = 1pean.pgm.oct
file = 1seb.pgm.oct
octave:3>  who

*** currently compiled functions:

columns  fr_load  isempty  mean     printf   reshape  rows     strcmp

*** local user variables:

depth   dim     mmm     nbfoto  y

octave:4>  dim
dim =

  100  100

octave:5>  yc=y-ones(dim(1)*dim(2),nbfoto)*depth/2;
octave:6>  imshow(reshape(yc,dim(1),nbfoto*dim(2)))
octave:7> % on obtient la galerie d'image normalisé
octave:7> v=cov(yc)
v =

  730.18  461.69  534.37  570.64  507.03
  461.69  739.02  457.75  449.85  425.81
  534.37  457.75  674.33  528.90  488.97
  570.64  449.85  528.90  804.26  598.13
  507.03  425.81  488.97  598.13  917.86

octave:8> [b,l]=eig(v)
b =

   0.711632   0.273072   0.432817   0.171945   0.449573
   0.054311  -0.197444  -0.631212   0.632283   0.399820
  -0.669828   0.513801   0.252685   0.199108   0.428772
  -0.190191  -0.761204   0.361913  -0.161553   0.476777
   0.076079   0.207424  -0.468413  -0.710570   0.476305

l =

   164.13221     0.00000     0.00000     0.00000     0.00000
     0.00000   196.24860     0.00000     0.00000     0.00000
     0.00000     0.00000   297.24862     0.00000     0.00000
     0.00000     0.00000     0.00000   415.25215     0.00000
     0.00000     0.00000     0.00000     0.00000  2792.76171

octave:9>  t = yc*b;
octave:10> imshow(reshape(t,dim(1),nbfoto*dim(2)))
octave:11>  who -variables

*** local user variables:

b        depth    dim      l        mmm      nbfoto   t        v        y
yc
octave:12>  clear  b;clear  depth;clear  dim;clear  l;clear  mmm;clear
nbfoto;clear v;clear y;clear yc;
octave:13> who -variables
```

```
*** local user variables:

t

octave:14>  % on passe à la personnalisation
octave:14> %  un usager veut "s'abonner" au système
octave:14> load 1mat.pgm.oct
octave:15> who

*** currently compiled functions:

colormap   fr_load    imshow     mean       rows
columns    gray       is_matrix  printf     saveimage
cov        image      isempty    reshape    strcmp

*** local user variables:

X     dims  ncol  t

octave:16> size(X)
ans =

  100  100

octave:17> yc=reshape(X,dims(1)*dims(2),1);
octave:18> size (yc)
ans =

  10000      1

octave:19>  yc=yc-ones(dims(1)*dims(2),1)*127;
octave:20>  vprime=cov(t,yc)
vprime =

   128.699
   -31.620
   -67.140
    93.068
   757.393

octave:21>  % on a la clésignature de 1mat.pgm.oct depuis t
octave:21>  % on inscrit ce vprime en carte
octave:21> % il faudra creer une fonction adaptateur
octave:21> % pour mettre en forme cette matrice dans
octave:21> % un format lisible par ZeitControlBasic par exemple
octave:21> %
octave:21> clear X; clear dims; clear ncol;clear yc
octave:22>  who -variables

*** local user variables:

t       vprime

octave:23>  % on passe a l'authentification
octave:23>  % on laisse t la cle publique
octave:23> % et vprime qui est lue depuis la carte à puce
octave:23> %
octave:23> % on prend un cliché de l'usager
octave:23> load 1mat2.pgm.oct
octave:24> who

*** currently compiled functions:
```

```
          colormap   fr_load    imshow     mean       rows
          columns    gray       is_matrix  printf     saveimage
          cov        image      isempty    reshape    strcmp

          *** local user variables:

          X       dims    ncol    t       vprime

          octave:25> X=reshape(X,dims(1)*dims(2),1);
          octave:26>  yc=X-ones(dims(1)*dims(2),1)*127;
          octave:27>  vseconde=cov(t,yc)
          vseconde =

               97.1495
               -9.3351
              -26.3722
              -27.5079
             1276.9406

          octave:28> vprime
          vprime =

             128.699
             -31.620
             -67.140
              93.068
             757.393

          octave:33> % essayons de trouver des facteurs
          octave:33> vprime./vseconde
          ans =

             1.32475
             3.38721
             2.54588
            -3.38334
             0.59313

          octave:34> vseconde./vprime
          ans =

             0.75486
             0.29523
             0.39279
            -0.29557
             1.68597

          octave:35> % ceci est plus interessant, on repere que ce vecteur
          octave:35> % "d'erreur" approx. (représentant les facteurs communs entre
eigenfaces respectifs du visage authentique et du visage à authentifier
          octave:35> %) contient :
          octave:35> %
          octave:35> %  3 valeurs comprises (dans l'absolu) 29.5%<x<39.2%
          octave:35> %  1 valeur supérieure à 75%
          octave:35> %     elle correspond au visage mineur
          octave:35> %  1 valeur supérieur à 168%
          octave:35> %     elle correspond au visage majeur
          octave:35> quit
```

Le modèle mathématique du système V2

Projet industriel : système biométrique de reconnaissance

Certifier :

$\alpha, K, c \quad (\boldsymbol{a})$
$t, l \quad (\boldsymbol{b})$

$G(t) \rightarrow \alpha \ ; \ \exists t \neg \emptyset$
$t * \alpha = K$

l

K

Personnalizer:

$k, K', l, c' \quad (\boldsymbol{a})$
$t', B, b, K \quad (\boldsymbol{b})$

$\varphi(K, rand()) \rightarrow l \ ; \ \exists K \neg indet$
$\Phi(K, l) \rightarrow K'$
$f(t', K') = k \ ; \ \exists t' \neg \emptyset$

$C(l, K) \rightarrow c \ ; \ \exists l \neg indet$

$C(l, K) \rightarrow c'$

c ?= c'

L	PublicKeys
l	K
l'	K
...	...

B,b

K'

Authentifier:

$r, k' \quad (\boldsymbol{a})$
$k, B, b, t'', K' \quad (\boldsymbol{b})$

$f(t'', K') = k' \ ; \ \exists t'' \neg \emptyset \quad (\boldsymbol{c})$

$\exists k \wedge k \neg \emptyset \ ; \ (b.k < k' < B.k) \rightarrow r \quad (\boldsymbol{c})$

$(\boldsymbol{a})$ *Entity calculed, output data*
$(\boldsymbol{b})$ *Entity input data (user input, network transmission)*
$(\boldsymbol{c})$ *Card microchip secured operations (i.e card stores k)*

α : *eigenvalue* $\quad l$: *hazard*
$G(t)$: *eig* $(v) \quad C(,)$: *certificate generator*
K : *eigenfaces, synthetic faces (public key)* $\quad c, c'$: *certificates*
t : *picture* $\quad t'$: *authentic picture* $\quad t''$: *suspicious picture*
k : *authentic private key* $\quad K'$: *authentic public key*
B, b : *superior, inferior limit*
φ : *hazard generator* ($rnd()$: *pseudo−hazard generator*)
Φ : *public key generator*
k' : *private suspicious key* $\quad r$: *truth value.*

Projet industriel : système biométrique de reconnaissance