University of Poitiers,

Master in Information and Communication Systems, Emphasis in Network Technologies

First Year Training Project Report
June 10th, 2015

Algorithmic tools and software for capturing facial actions.

Students: Ali Toilha
Guy-Florent A. Sadeler
Viviane A. Basse

Supervisors: Pascal Bourdon
David Helbert
Christian Chatellier

Contents

Introduction

At the University of Poitiers, students in Master (professional and research career) of Information and Communication Systems, Emphasis in Network Technologies field have to carry out a project during their first year in order to complete the training. This is the main reason we wrote this report of project.

In fact, several topics were proposed to us and we choose the facial recognition for its multiplicity and variety in term of application fields, such as high security applications, remote monitoring and control of access etc. Indeed, this field is closely linked with human vision reproduction researche and the interaction between a machine and a human, which is a fascinating and booming subject.

Facial recognition consist in identifying a person with a picture of his face. It appears to be a quite active field in Vision from computer and Biomety. Recognition with an individual face is a skill of Biometry which is still to improve. Indeed, the acquired (captured) picture represents variation much higher than other characteristics (makeup, presence or absence of glasses, aging and expression of emotion). The method of face recognition is sensitive to the variation of the light and change the position of the face in the image acquisition. But still the system obtained is not yet able to adapt itself to certain kinds of variations on a face.

In fact, achieve the two prototypes necessary for our analysis we adopted the following outline anouncing this report. First, we define the project environment and aims at first, then we develop the state of the art methods to be programmed, as it were the theoretical concept. Then we establish a technical report is to explain the use of our prototypes made and finally we leave the analysis of the use of these prototypes to the comparison of our results.

Chapter 1

Presentation of the project

1.1 Context and scope of work

1.1.1 Context

Previously carried out at the SIC Department of XLIM laboratory, the project we are in charge of implements "algorithmic tools and capturing software for facial recognition". This project interest was also interactions with and between users of a video game with an educational aim ("Serious game") by automatically animated avatars and experienced analysis difficulties, as well as the development of a software library for developers of video games on smartphones.

Facial recognition is often used as a tool to secure devices or control the use of applications. This technique both helps adults secure their systems and devices and parents limit and control access to systems for their kids. The main idea was to improve parental control of tablets for children's use. This could be helpful for parents wishing to use facial recognition to limit their children's use past a certain time of a day.

1.1.2 Scope of work

This project aims to develop a facial capture software program (face recognition) in XLIM-SIC laboratory research work in collaboration with a company based in Lyon operating in the field of manufacturing tablets for children . The required results at the end of this project are: facial recognition software program with

- Eigen Faces;
- Fisher Faces.

Those will be programmed with multiple programming languages (Python, C ++). This project is performed with the research professors of the Fundamental Faculty of Sciences at the University of Poitiers: Pascal Bourdon and David HELBERT who supervises the project

1.2 Objectives

The work required for this project is the development of a share recording software for the recognition of facial expressions for the identification of a face from an ID list in a home or for parental control. Today innovation and the latest technologies are growing increasingly and allow the interactivity between human-machine to be maximal. Research on facial expression is fundamental in many applications. Facial recognition takes place in three stages, namely:

- Face detection
- Extraction and normalization of facial features
- Identification and / or verification

The main difficulty in face recognition is that there are no two identical faces. Thus, each individual is unique and will be marked by gender, ethnicity, age or his haircut, but also by the shape, size and arrangement of the elements of the face.

This project has an educational goal because it contributes greatly to our engineering multimedia training, allowing us to put into practice the theories studied in the various teaching modules of our maste'sr degree (Image processing, tool and scientific computation, Algorithms for multimedia, random signal processing) but also by completing them. This project can be seen as a complement to our training.

1.3 Deadlines

Deadlines	Deliverables
April 2	Users requirements presentation + handouts
May 13	State of the art Theoretical
June 11	Official delivery (technical documents, codes, ...)
June 18	Project presentation

Table 1.1: **Deadline Table**

Chapter 2

State of the art

2.1 Facial Recognition

The challenge of face recognition can be formulated as followed : with one or several images of a face, the goal would be to find or check the identity of a person by comparing his face to all the face images stored in a database. By the way this skill remains the most acceptable because it more suits with what human beings use in visual interaction; and compared to other methods, the face recognition seems more advantageous, in fact it is a non-intrusive method, in other words it does not require the cooperation of the subject, and a moreover the sensors used are cheaper.

2.1.1 Facial recognition process

Any facial recognition process must take into consideration several factors that contribute to the complexity of its task, because a face is a dynamic entity which constantly changes under the influence of several factors. A facial recognition system is generally divided into the following steps (see the figure):

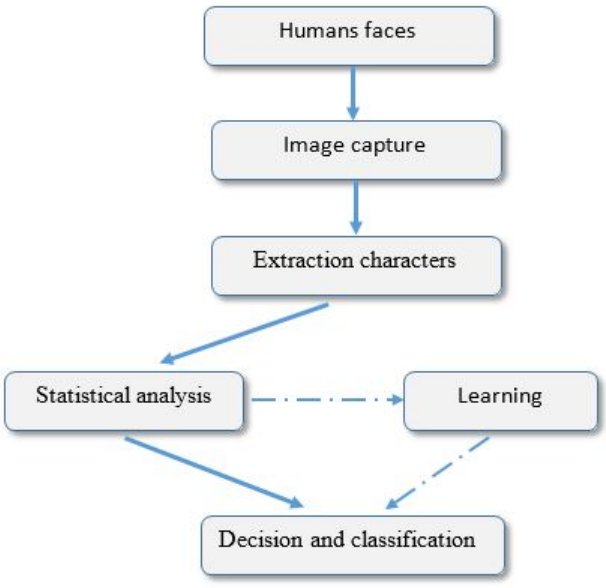

Figure 2.1: **Facial recognition system process**

Facial recognition is facing the following problems:

- Change of pose ;
- Light Variations ;
- Variations of expression, age ;
- Partial occultation of the face (concealing).

These variations are the most difficult because the variations in the appearance of a person face according to different pose or light conditions are often far more important than the variation between face images of two different individuals acquired under the same conditions. This explains why pictures should be taken in specific conditions so that facial recognition can be efficient.

2.1.2 The methods used for face recognition

Facial recognition methods can be classified into two broad categories: local and global methods. Amongst those methods, main ones will be presented thereafter.

2.1.2.1 Global methods

Global methods are based on well known techniques of statistical analysis. In these methods, face images (which can be shown as matrices of pixel values) are used as input of the recognition algorithm and are generally transformed into vectors, which are easier to handle. The main advantage of global methods is that they are relatively quick to set up in. However, they are very sensitive to variations of illumination, pose and facial expression.

The main existing methods are:

- The Principal Component Analysis (PCA) : EigenFaces
- The LDA (Linear Discriminant Analysis) Algorithm : FisherFaces

2.1.2.2 Local methods (Geometric)

The local methods include transformations applying to specific areas of the image, usually around characteristic points (corners of the eyes, mouth, nose, ...). Therefore, they require a priori knowledge on images. These methods are more difficult to implement but are more robust to the problems due to variations of illumination, pose and facial expression. The main existing methods are:

- EBGM (Elastic Bunch Graph Matching);
- Modular Eigenface;
- Hidden Markov Method.

But in fact, our aim on this project will be obviously to use both main global methods.

Both methods that we will present are using a common training algorithm steps that are :

- Preprocessing of training image set
- Normalization and estimation of mean image
- Use of PCA/ LDA

PCA/ LDA are statistical tools used to implement facial recognition method. For instance the use of PCA is divided into two steps :

- The determination of the input image weight from projecting input image into the face space and by multiplying the resulted vector to eigenfaces of the database.
- A Comparison of results with metrics such as euclidian distance.

2.2 Eigenfaces

2.2.1 Presentation of Eigenfaces

The Eigenface approach began with a search for a low-dimensional representation of face images by Sirovich and Kirby in 1987. It is the first method considered as a successful technique of face recognition. The Eigenface method uses Principal Component Analysis (PCA) to linearly project the image space to a low dimensional feature space and it is the name given to a set of eigenvectors when they are used in the computer vision problem of human face recognition

2.2.2 Procedure

A set of eigenfaces can be generated by performing a mathematical process called Principal Component Analysis (PCA) on a large set of images.

To create a set of eigenfaces, one must:

- Load a training set of face images. The pictures of the training set should have been taken under the same lighting conditions, and must be normalized to line up the eyes ,mouths and other features.
- Compute the average image : add each columns of the matrix T and dividing the previous obtained vector by the number of input images.
- Subtract the mean from matrix T to obtain matrix S
- Calculate the covariance matrix S.
- Calculate the eigenvectors and eigenvalues of the covariance matrix S. Each eigenvector has the same dimensionality as the original images. The eigenvectors of the matrix S are called eigenfaces.
- Choose the principal components. The number of principal components k is determined arbitrarily by setting a threshold ϵ on the total variance.
- Determinate the input image weight determination from projecting each image
- Each image is represented by a vector which is used to reconstruct the images. We then save the average image, eigenfaces and the projection (weight) of images.

This ends the training part of the implementation of eigenfaces and shows the skills used.

2.2.3 EigenFaces flowchart

The flowchart we have to use is divided into two basic parts: the learning phase and the identification phase where the Euclidean distance is used to calculate the difference between the weight of the image to be identified and the database images, then the program displays the nearest.

But retain before these two major steps,we have pretreatments and it's the phase which is carried out :

- The selection of the learning base ;
- Reading images ;
- The conversion of grayscale images ;
- Resizing images ;
- And finally the application of histogram equalization.

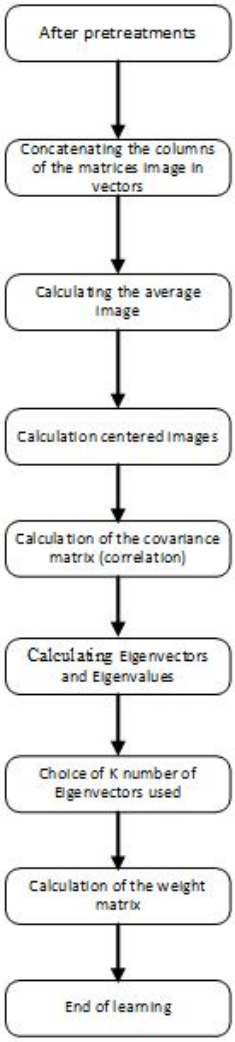

Figure 2.2: **EigenFaces flowchart learning phase**

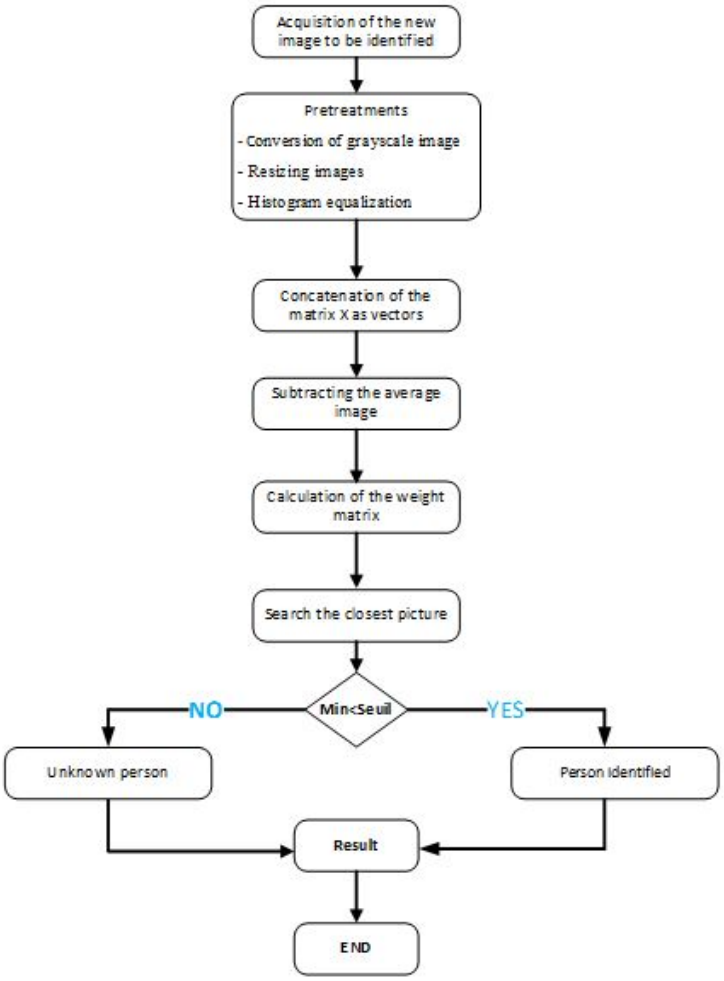

Figure 2.3: **EigenFaces flowchart identification phase**

2.2.4 Benefits and deficiencies

2.2.4.1 Benefits

Eigenface provides an easy and cheap way to realize face recognition in that:

- Its training process is completely automatic and easy to code.
- Eigenface reduces statistical complexity in face image representation.
- Eigenface can used large databases.

2.2.4.2 Deficiencies

The deficiencies of the eigenface method are:

- Very sensitive to lighting, scale and translation;
- Eigenface has difficulty capturing expression changes.

The most significant eigenfaces are mainly about illumination encoding and don't provide useful information regarding the actual face.

2.3 Fisherfaces

Eigenface method uses PCA for dimensionality reduction, which yields projection directions that maximize the total scatter across all classes of images. This projection is best for reconstruction of images from a low dimensional basis. However, this method doesn't make use of between-class scatter. The projection may not be optimal from discrimination for different classes.

While this is clearly a powerful way to represent data, it does not consider any classes and so a lot of discriminative information may be lost when throwing components away.

The Fisherface method is an enhancement of the Eigenface method that it uses Fisher's Linear Discriminant Analysis (FLDA or LDA) for the dimensionality reduction. The LDA maximizes the ratio of between-class scatter to that of within-class scatter, therefore, it works better than PCA for purpose of discrimination. The Fisherface is especially useful when facial images have large variations in illumination and facial expression.

This projection maximizes the ratio of between-class scatter to that of within-class scatter. The idea is that it tries to "shape" the scatter in order to make it more reliable for classification.

2.3.1 Linear Discriminant Analysis (LDA)

The Linear Discriminant Analysis (LDA) is used to find the linear combination of characteristics that better separate object or event classes. The resulting combinations can be used as a linear classifier, or generally in reducing characteristics before the posterior classification.

LDA is closely related to the PCA, because both seek the linear combinations of the variables that better represents the data. This statistic skill explicitly to model the difference between data classes unlike the PCA which does not take into account the differences between classes.

2.3.2 LDA for recognition

The LDA by recognition algorithm is divided into two phases, one for the calculation of person models called system learning phase and the other which is to recognize a person thanks to registered models called test stage.

2.3.2.1 Learning phase

As in the PCA, it gathers the images of the learning database in a large image matrix T where each column represents a image listed Ti, then the average image is calculated. For each class C, the average image is calculated. Each image Ti of each class Ci is then refocused in comparison of the average. This produces a new image. Then we proceed at calculation of the different scatter (dispersion) matrixes named as followed:

- The Intra-class Distribution Matrix;
- The Inter-class Distribution Matrix;
- Total Dispersion Matrix.

After we have defined the different dispersion matrixes, we must find the best projection that maximizes the intra-class dispersion on its matrix while minimizing inter-class dispersion, also on its matrix.

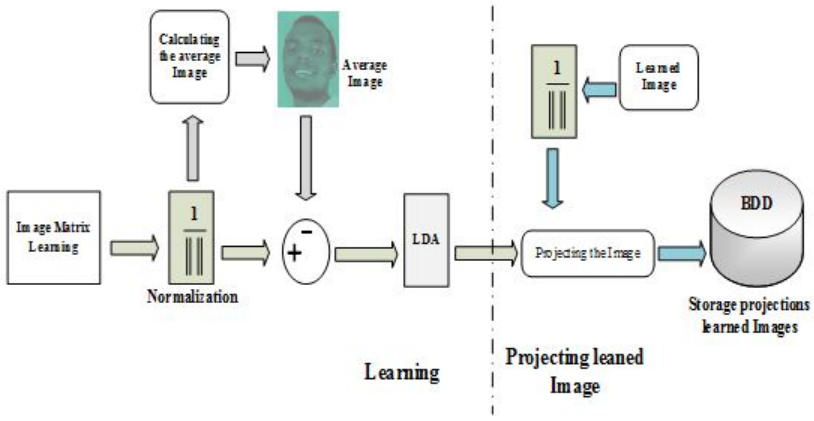

Figure 2.4: **Learning phase of a face recognition system using a global method**

2.3.2.2 Test phase or test stage

Once the optimal projection that maximizes the found intraclass dispersion, the same pattern as the PCA on the projection of learned image and the projection of a test image is applied. Then we project the test image in the Fisher space. We compare the models obtained in learning phase. The comparison is made by calculating the distances (for instance, one can use calculation of the Euclidean Distance) between models and the test vectors and a decision rule is used to classify people.

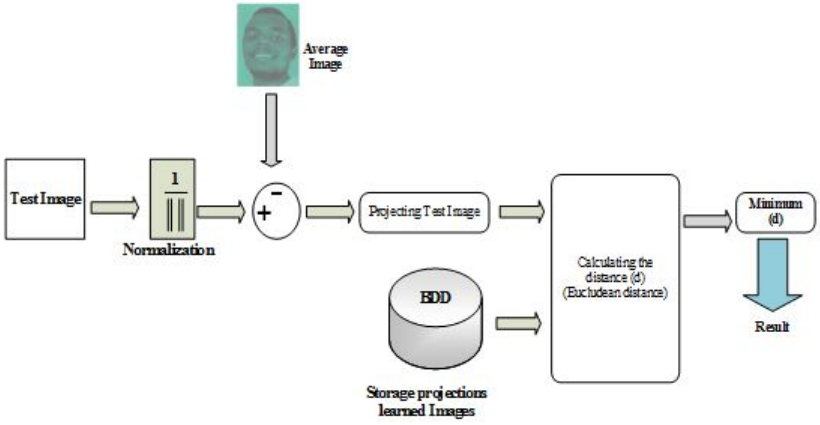

Figure 2.5: **Test phase of a face recognition system using a global method**

2.3.3 Fisherfaces flowchart

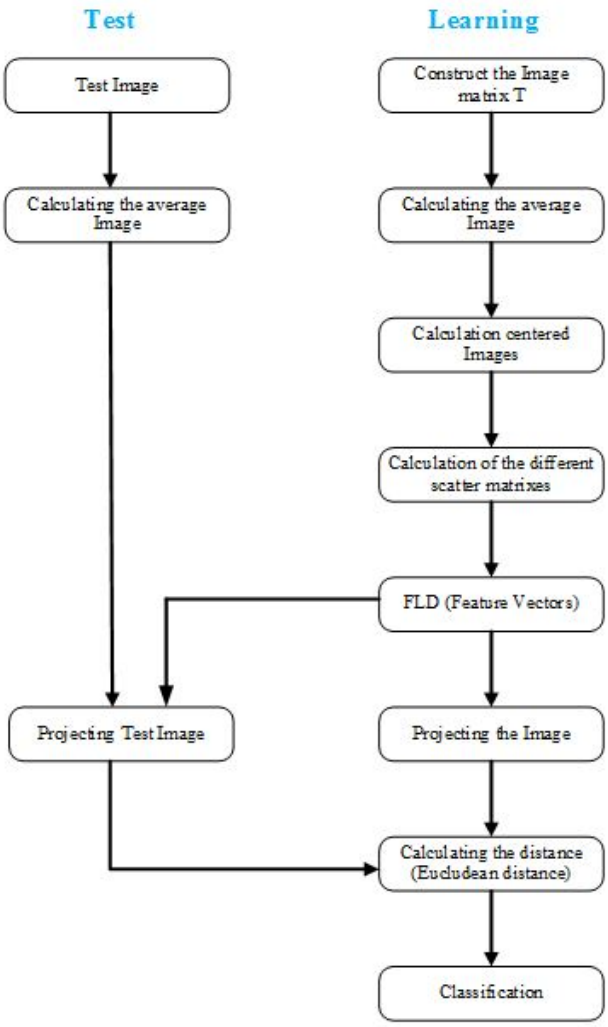

Figure 2.6: **FisherFaces flowchart**

2.3.4 Benefits and deficiencies

The use of the LDA method for face recognition afford the following advantages:

- Maximizing inter-class scatter ;
- Reduction of intra-class scatter ;
- The method of Fisherfaces solves the problem of robustness to changes in pose, and facial expressions.

Despite these advantages, in the literature a serie of negative spikes still exists as:

- Costly (heavy) in computation time ;
- Costly (heavy) in memory space ;
- Makes poor results when the number of training images is great.

Chapter 3

Organization

3.1 Project Management System

In order to achieve our project on schedule we had to define steps to optimize the management of our time and to avoid delay on deadline. During early months at the launch of project, we started by writing the user requirements, which is a document etablished according to the need of the product owner. The user requirements allowing all users to orient the project progress and to keep in view the expected objectives. So this is a writing task. This user requirements was presented to the product owner and before an academic jury. This was about getting validation on user requirements of the product owner so that we make sure that his needs are clearly understood by us.

Then we had, thanks to Mister Chatelier's training a project manangement system called scrum. This system was about detailling all the tasks there was to complete and specifically the sprints.
In this part, we will describe the organization of our work throughout the project and will present tools used to the project management.

For the project management,we used the Backlog and the Trello.

3.1.1 Backlog

Thanks to the training given by Mister Christian Chatellier, we define the implementation of the project management system.
This system includes the backlog writing which listseach individual task there is to complete during the project and affords us to define every sprints (knowing that a sprint represents a week of work).
In the backlog all tasks are classified by priority and if a task contained in the backlog does not contribuate to the project objectives ,we must remove it

SPRINT EN COURS 4 **BACKLOG PRODUIT**

Sprint	Catégorie	Sous catégorie	Nom / Description	Importan	Estimatio	Critères de Vérification	Status
1		redaction	logigramme pour les eigenfaces	60	1,5	relecture par un autre membre (concordance entre la schematisation et l'etude theorique)	En cours
1		redaction	logigrammme pour les fisherfaces	60	1,5	relecture par un autre membre	En cours
1		redaction	synthese et revision de l'etat de l'art	60	1,5	relecture par tous les membres du projet	à faire
1		redaction	documentation sur la reconnaissance faciale	60	2	relecture par le chargé de la tâche et un autre	Terminée
1		redaction	documentation sur les eigenfaces	60	2	relecture par le chargé de la tâche et un autre	Terminée
1		redaction	documentation sur les fisherfaces	60	2	relecture par un des charges de la tâche et un autre membre (orthographe,chronologie,véracité des	En cours
1		présentation	reunion avec le superviseur	55	1,5	Validation auprès du superviseur	à faire
		Recherche	documentation sur python et	50			à faire

Figure 3.1: **Backlog**

In the backlog ,we have many parameters:

- Sprint:current sprint of a moment(a sprint correspond to a week)
- Subcategory : category of a task (redaction,research,programming or meeting with the product owner)
- Name/description: description of a task
- Importance : priority of task
- estimation : time or duration for a task
- criteria for verification
- Status : indicates the status of a task(doing,done)

Note that every friday we change the priority of certains tasks for praparing the next sprint.

3.1.2 Trello

3.1.2.1 Presentation

The second tool used is Trello Trello is a project management tool allowing to list all project tasks.It is composed by many cards.
With Trello we launch the scrum every morning by a meeting between members of the group projet.It enables to see the progress work for all member of the project.
our Trello is divides into five cards:

- To do :list of all tasks that we have to do for the week
- Doing: list of current tasks (Task for a day)
- Wait :tasks waiting for very precise reasons
- Test and Validation : task done but not yet tested
- Done : Completed Tasks

Note also that the project is divided into several weeks each corresponding to a Sprint.

3.1.2.2 Sprint

There we present the four sprint of the project :

- Sprint 1
  For this sprint we complete a second part which is the writing of the state of art. A document reporting the theoretical aspect to be developped in the project but also the explanation of the methods used and the algorithms to be programmed.During this sprint all intended tasks have been completed

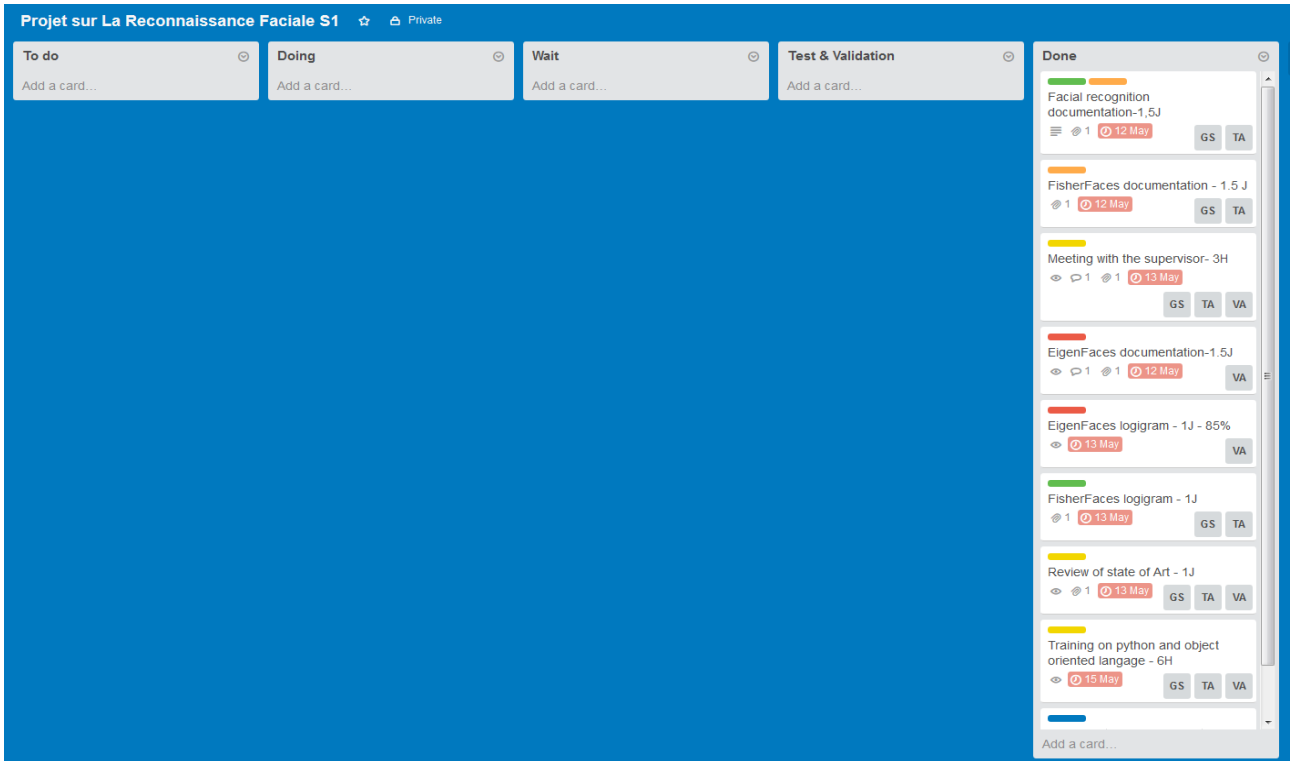

Figure 3.2: **First sprint**

- Sprint 2
  At the sprint 2, we did a literature search to deepen the appearance of algorithmic methods and presented the results of our research so that we all stand on the same point of view.We begin the prototype for eigenfaces method in python.

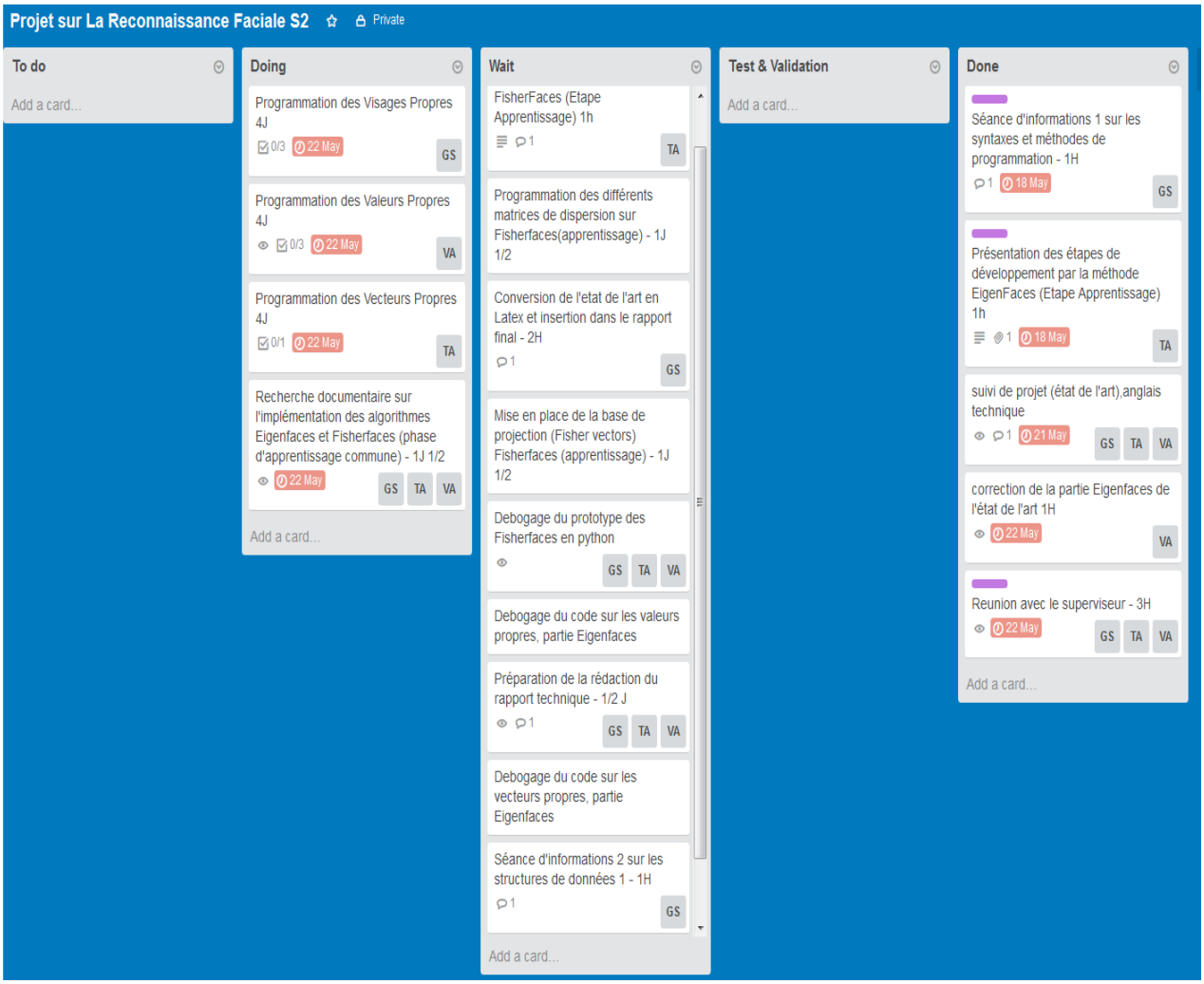

Figure 3.3: **Second sprint**

- Sprint 3
  This sprint consists in finishing the previous tasks started in later sprint about programming eigenfaces method. But added to those tasks we scheduled in the writing tasks. By the way some of the programming tasks uncompleted are renewed in Sprint 4.

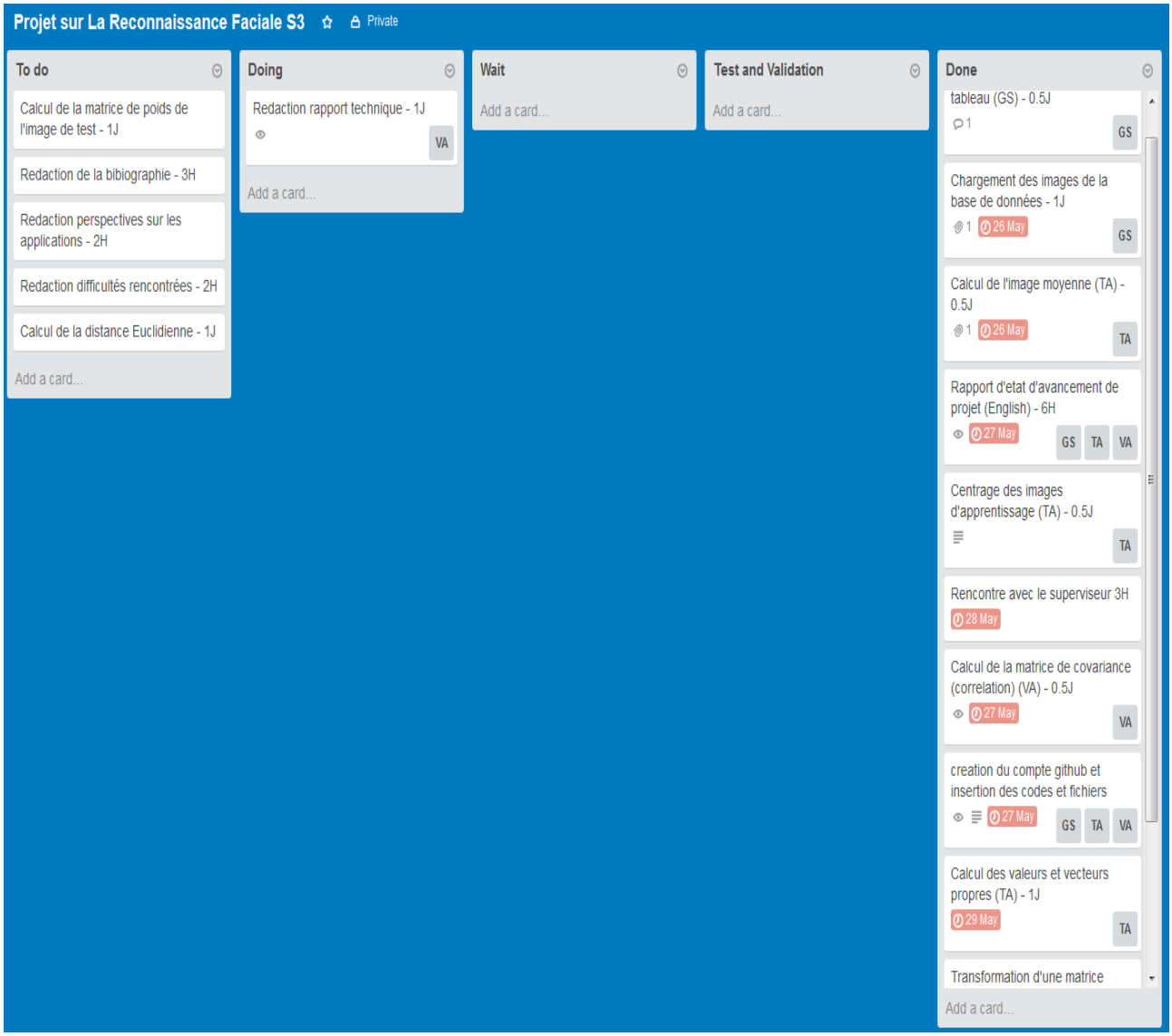

Figure 3.4: **Third sprint**

- Sprint 4
  For this sprint, we started by programming Fisherfaces method but did not stop us to progress on the eigenfaces method.We also achieved some writting tasks.

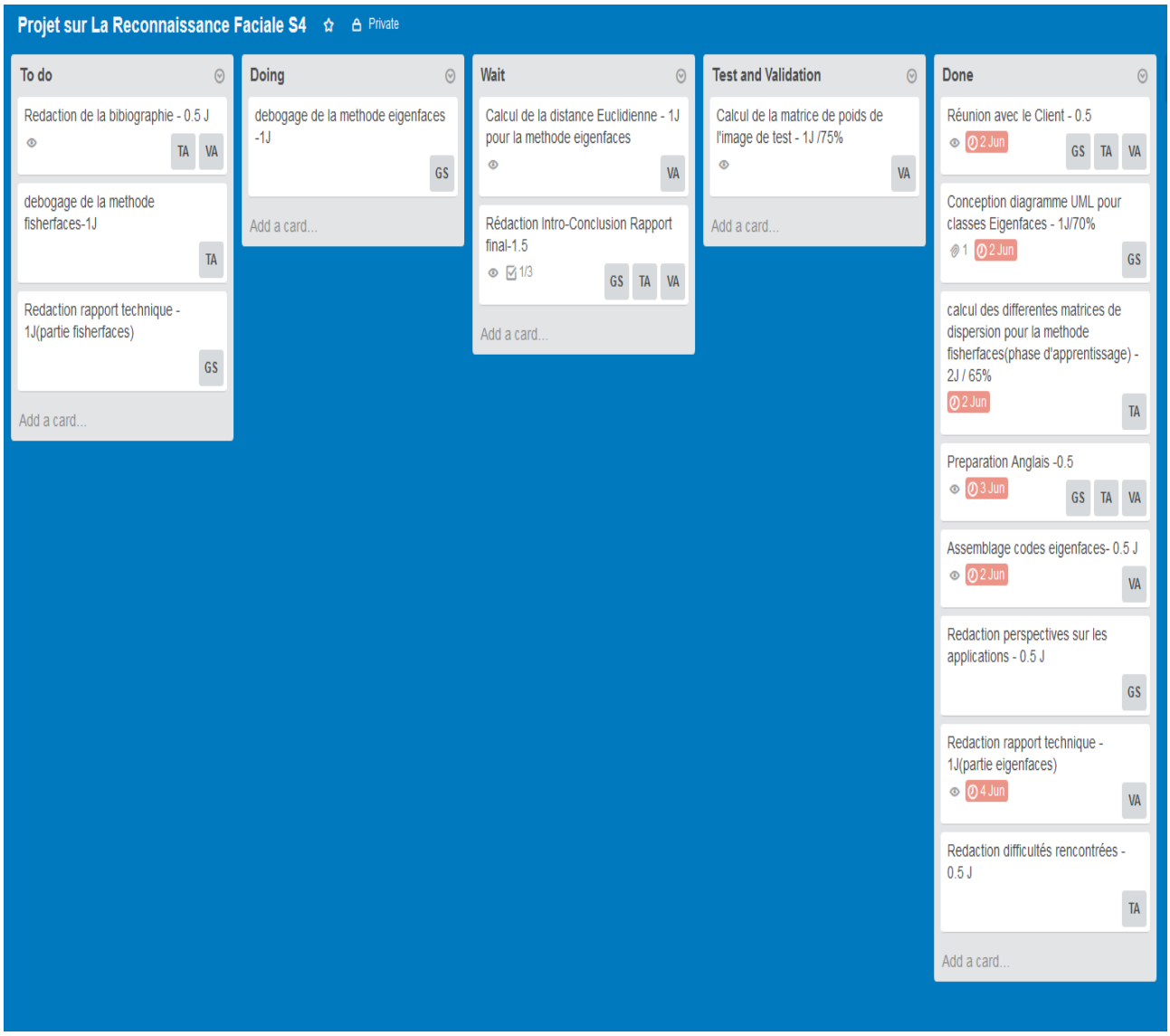

Figure 3.5: **fourth sprint**

3.2 Team Project

The sponsor of this project is M. Pascal Bourdon working for XLIM-SIC Laboratory in Poitiers as a researcher. The results will be used by the Laboratory.

The project team includes :

Supervisors:

- Pascal BOURDON as Supervisor and Product Owner;
- David HELBERT as Academic Training Principle;
- Christian CHATELLIER as Project Management Supervisor.

Students in charge:

- Viviane Arame BASSE as Communication Manager
- Guy Florent A. SADELER as Technical Manager
- Ali TOILHA as Project Manager

The End users of this project are:

- XLIM-SIC Laboratory,
- RTMA Training of the University of Poitiers for tutorials,
- Product owner and any potential company requiring facial recognition technology.

Chapter 4

Technical report

4.1 Guide on Eigenface prototype

In this section, we will explain the implementation of eigenface method with the differents expressions used to program

Using mostly PCA (Principal Components Analysis), eigenfaces method is based on the calculation of eigenvectors and eigenvalues. The algorithm consists of two steps:

- the learning phase : in which the protoype define a projection space with several person images;
- and the identification phase : which consist in projecting the face of the person one needs to recognize from the database.

Those steps are presented as followed :

4.1.1 Learning phase

The learning phase can be divided into several steps:

- Step 1 : It is necessary to have an image database containing M images taken in good conditions. In our case we used the AT&T picture database contains 400 images. The database includes 40 subjects and each subject has 10 differents images of himself so that he can be easily recognised in differents conditions (with glasses, a hat, a smiling face for instance...). All these images are the basis for learning. Each image is a matrix.
- Step 2 :Each image matrix is converted into vector.

$$\begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{pmatrix} \longrightarrow A = \begin{bmatrix} a_{11} \\ \vdots \\ a_{nm} \end{bmatrix}$$

- Step 3 : Each image vector as previouly obtained becomes columns of a single matrix Γ. Every single image Γ_i of the database is then in that single matrix and all of them are sorted from a subject to another.
  a:the subject 1 and z the subject n

$$\Gamma = \begin{pmatrix} a_{11} & \cdots & z_{11} \\ \vdots & \ddots & \vdots \\ a_{nm} & \cdots & z_{nm} \end{pmatrix}$$

- Step 4 : Calculate the average Ψ of all images

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

- Step 5 : Substract the average image from each image

$$\varphi_i = \Gamma_i - \Psi$$

- Step 6 : Calculate the covariance matrix S

$$S = \sum_{i=1}^{M} \phi_i * \phi_i^T = A * A^T, A = (\phi_1, \cdots\cdots\cdots, \phi_M)$$

- Step 7 : After the covariance matrix ,eigenvalues and eigenvectors are computed. Eigenvectors are then sorted by decreasing order.

$$S * e_i = \lambda_i * e_i$$

$$\text{Formulas : } \begin{cases} e_i = A * v_i & \text{eigenvectors} \\ \lambda_i = \mu_i & \text{eigenvalues} \end{cases}$$

4.1.2 Identification phase

The identification phase includes two steps and will help to recognize an input image in the image database.

- Step 1 : compute projection vectors

  $w_k = e_k^t * (\lambda_i - \psi)$

  The projection vectors are called "weight vectors " and form a single matrix which will help for compute Euclidean distance. It also will help to find the class for an input image.

- Step 2 : compute the Euclidian distance

This step in similar to the identification phase of Fisherfaces method.

4.2 Guide on Fisherfaces prototype

This section is détailling differents steps on fisherfaces

4.2.1 Learning phase

The first part is also programmed into two steps the first step that is the learning phase is almost similar to fisherfaces' one. This similarity is onto the following common tasks:

- Loading of 10 images for every 40 subjects from the chosen database repertory;
- Processing on images to get the mean image and center the images.

Once that common steps with eigenfaces are implemented or set up, we had to develop the LDA algorithm. This step can be describe as followed :

Calculation of the scatter matrices

- Step 1 : We first get our "y" list gathering the numerotation of images according to the subject they belong to. Then this "y" vector involved numbers from 0 to 39 since we have 40 subjects and each of this number are repeated ten times since every single subject has ten pictures. With the command "$c = np.unique(y)$", we get the same vector in "c" of numbers without them repeated tenth by sorting elements of "y" and eliminating duplicates. Those numbers return the classes necessary to compute scatter matrix.
- Step 2 : The second step will be to calculate the scatter matrices :

$$Sw = Sw + np.dot((Xi - meanClass).T, (Xi - meanClass))$$ Within classes scatter matrix (Sw)

$$Sb = Sb + n * np.dot((meanClass - average).T, (meanClass - average))$$ Between classes scatter matrix (Sb)

Calculation of fisherfaces

- Step 3 : We then apply the LDA algorithm to the previous reduced parameters We need to find an optimal projection basis which both maximize the within dispersion relative to its matrix Sw and minize the between dispersion relative to its matrix Sb.
  In other words, it consists in finding the "W" factor that maximize the Fisher optimizing Criteria :

$$\begin{aligned} W_{\text{opt}} &= \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|} \\ &= [\mathbf{w}_1 \mathbf{w}_2 \ldots \mathbf{w}_m] \end{aligned}$$

- Step 4 : We choose to use the PCA method to solve the problem we faced while trying to get W. The problem is about having matrices of different size. Then to resize them and be able to calculate W, PCA has been quite helful.

4.2.2 Identification phase

The recognition of image from the basis obtained is similar to the previous method and take into consideration the calculation of distance such as euclidian distance, cosine .... We choose the euclidian distance.

Chapter 5

Implementation

5.1 Configuration Management Guide

In this section, we define a guide helping us programming and defining our structures in a homogene in order to easily read each other and to help the integration of our separate codes. This, since we programmed on the same prototype seperately. This document help us have homogenized codes on python.

Structure code : For every languages used, we will prefer the definition of code in two parts:

- First part: definition of functions and procedures or methods;
- Second part: Main function.

This process reduces the recall of certain calculations and allow to have a better visibility of code. It is necessary follow the order of definition for compilation.

Comments : Each different parts should have a brief description before it is edited. functions and procedures have a functional description; tvariables are commented after their definition.

Definition of variables :

- Variables are defined at the beginning of each entity (functions, procedures, classes, hands etc ...) except possibly in object-oriented programming;
- The name of a variable, whether local or global, is the simplest possible and defined in lowercase;
- we will take care to separate input variables from those output for better visibility.

Definition of functions, procedures or methods :

- As with object-oriented programming (JAVA) we will take for a function, at least two significant words with the first word in lowercase letters and the first letter of the second word capitalized. It provides for example in case of two words "calculateMatrix" and three words "calculateScatterMatrix".

Definition of classes :

- Classes are defined in capital letters(first letter), they are the superior entities after the Object class.

Definition of Main :

- Before each function call comment out by a significant title.

Identation :

- Carefully idente your code.

5.2 Implementation results

As said lately,for the implementation of eigenfaces method we used AT&T database composed of 400 images. The database consists of 40 subjects and each subject has 10 images.

We illustrate hereinater some images of the database :

Figure 5.1: **images of database**

All images of database are loading from directory by the function **read_img.py**(annexes). After that,the average of all images of the database will be calculated by the functiun **computeMeanImage_.py** and centered by **computeCenteredImage_.py**.
The average image is a vector and for saved it we needed to convert it into a matrix(annexes).

The illustration of average image is :

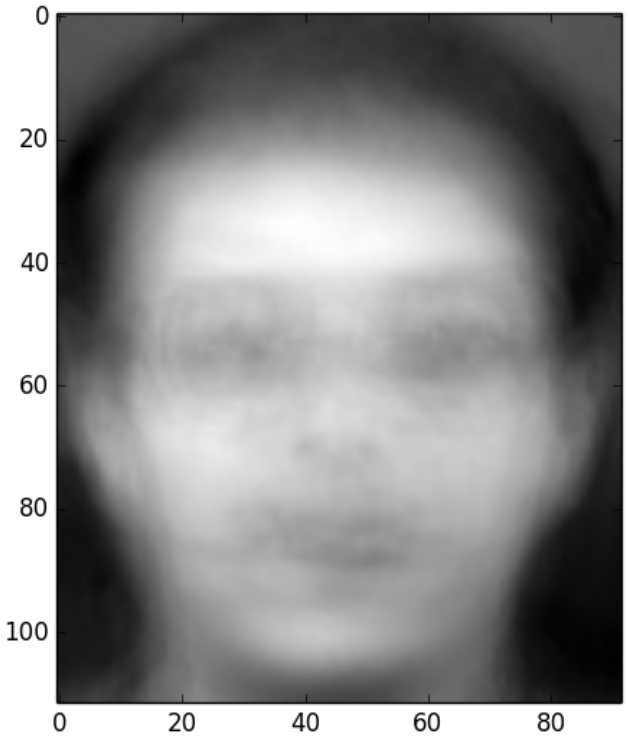

Figure 5.2: **Average image**

The average image is fluzzy because many images are used.

5.3 Integration into Oriented Programming Language

In order to create Objects from our prototype we designed the main classes presented as following :

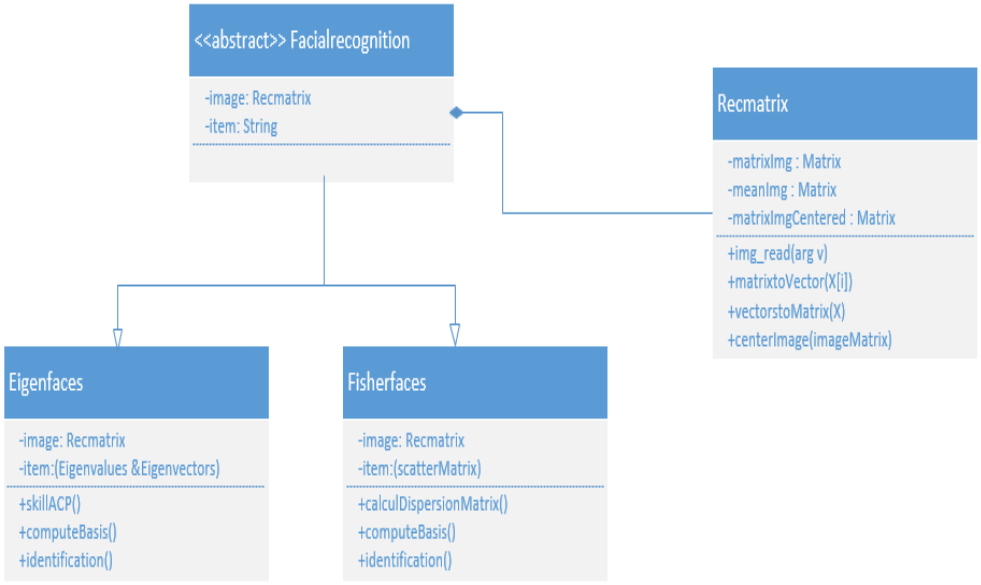

Figure 5.3: **UML Flowchart**

5.4 Prospects

Facial recognition develop tools very useful for people such as our first goals which were to improve parental control and to develop serious in order to help childrens with handicap. But so far, still there are some applications like high security, and ... All those applications require less human efforts, and then the use of this method are factor of unemployement and reduce responsibility from parents. On this project, we would have expected to have much more time to implement the methods in other way and to optimize much more the programs.

5.5 Difficulties faced and summary

5.5.1 Difficulties faced

When the project began, we set several objectives. To achieve them, we have established a schedule for work organized. But throughout the project, we have encountered some difficulties: technical difficulties and organizational difficulties.

The main difficulties are :

- implementation difficulties : because for programming methods, we had to define a unique way to structure the program for consistency in all codes. We have thus established a reference document to have uniform codes.
- understanding difficulties :everyone had understood in a different way from others the concept of the methods(eigenfaces and fisherfaces) for the implementation of our algorithm.
- Programming difficulties :we have different levels in programming. But that does not prevent us to program
- Using tools difficulties : everyone had to program a part of the code, so we needed a way for share and synchronize our codes. the tool was new and we took time to control this,
- management difficulties : as we have already said, we set goals and we had established a schedule for work in an organized way, but many secondary, unanticipated and unforeseeable tasks appeared.

However we faced all these difficulties, which allowed us to learn more about the two main domain of our project: images processing and programming language . Finally, these difficulties may be helpful for future work. These can be apprehended in another project

5.5.2 Summary

This project allowed us to gain maturity in the work and help us to understand the importance of schedule, organization, and discipline necessary to achieve a project. In addition, it also gave us the satisfaction of completing the project in a group. Each member benefits from the skills of other members. That share of skills and the adding skills we have gained make us think it is such a very positive experience to our training and future carreer. Because this experience has allowed us to figure out the complexity to manage a project, but also the difficulty of understanding and the importance of communication in a group.

Conclusion

The objectif of the project was to develop a facial capture software program (face recognition) in XLIM-SIC laboratory research work in collaboration with a company based in Lyon operating in the field of manufacturing tablets for children.

To achieve this project, we used two methods to implement: the eigenfaces and Fisherfaces. The eigenfaces using principal component analysis are a set of eigenvectors used in the field of vision to solve the problem of face recognition. The eigenfaces allow to reconstruct a subspace retaining the best eigenvalues while keeping much useful information. The Fisherfaces in turn use the LDA that analyze eigenvectors of the scatter matrix and aims to maximize the inter-class variations while minimizing the intra-class variations.

We have started with the implementation of eigenfaces method in python. For that it was necessary for us to have an image database. we used the ATT image database. However, some difficulties were encountered during programming which delayed us on objectives. That why we focused on eigenfaces method and we could not make a comparison of the results of both methods

Bibliography

[1] BETTAHAR Abdessettar and SABER Fathi. *MEMOIRE MASTER ACADEMIQUE, Extraction des caractéristiques pour l analyse biométrique dâĂŹun visage.* UNIVERSITE KASDI MERBAH OUARGLA Faculté des Nouvelles technologies de l information et de la communication, 15 Juin 2014.

[2] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces : Recognition using class specific linear projection, ieee transactions on pattern analysis and machine intelligence, vol. 19, no. 7,. JULY 1997.

[3] W Zhao. Belhumeur, JR Chellappa, and PJ Philips. Kriegman. Face recognition : A literature suvey. December 2003.

[4] Harry Chao. *Eigenfaces and Fisherfaces*, Mai 2010.

[5] SciPy developers. Numpy and scipy documentation, Mai 2015. http ://docs.scipy.org/doc/.

[6] Python Software Foundation. Python 2.7.10 documentation, Mai 2015. https ://docs.python.org/2/.

[7] CHELALI FZ and Mr DJERADI et Mme DJERADI. Mise en place d'un système de reconnaissance facial basé sur l'approche statistique analyse en composantes principales développée sous l'environnement matlab. 2005.

[8] Tal Hassner and Lihi Zelnik-Manor. Principal component analysis, eigenfaces and fisher discriminant.

[9] Houda Maâmatou. *MEMOIRE DE MASTER de recherche, Reconnaissance et mappage des expressions faciales vers un modèle virtuel.* Université de Sousse, Ecole Nationale dâĂŹIngénieurs de Sousse, 2012.

[10] Math Programming. Eigenfaces, for facial recognition, Mai 2015. http ://jeremykun.com/2011/07/27/eigenfaces/.

[11] Dewi Agushinta R. and Indri Septadepi. *Face Recognition System Using Eigenface Method based on Facial Component Region*, Mai 2010.

[12] safaribooksonline. Face recognition using eigenfaces or fisherfaces, Mai 2015. https ://www.safaribooksonline.com/library/view/mastering-opencv-with/9781849517829/ch08.html.

[13] Naotoshi Seo. Eigenfaces and fisherfaces.

Annexes

Before creating functions,we imported different libraries.

```
 7 import sys
 8 #import cv2
 9 sys.path.append("G:\\ProjetV1.1")
10 import numpy as np
11 import matplotlib.pyplot as plt
```

The module sys gives direct acces to the arguments of the command line.
The module numpy is a digital library providing efficient support for large multidimensional arrays,and high-level mathematical funtiuns as linear algebra,statistics ...
The module Matplotlib is a library programming language that combined with python library numpy and scipy is a powerfull tool for tracing and visualizing data.

5.6 Eigenfaces functions

In this part ,we will put the different functions and their explenations

- Function used to load images from directories

```
13 def read_img(path, sz=None):
14     c = 0
15     X,y = [], []
16     for dirname, dirnames, filenames in os.walk(path):
17         for subdirname in dirnames:
18             subject_path = os.path.join(dirname, subdirname)
19             for filename in os.listdir(subject_path):
20                 try:
21                     im = Image.open(os.path.join(subject_path, filename))
22                     im = im.convert("L")
23                     if (sz is not None):
24                         im = im.resize(sz, Image.ANTIALIAS)
25                     X.append(np.asarray(im, dtype=np.uint8))
26                     y.append(c)
27                 except IOError:
28                     print "I/O error({0}): {1}".format(errno, os.strerror)
29                 except:
30                     print "Unexpected error:", sys.exc_info()[0]
31                     raise
32             c = c+1
33     return [X,y]
```

This function has input the full path to load all the images of the database. It returns X which is a list of all the combined images and y which is the subscript i of all images for the subject i.

- Function used to transform matrix to vector

```
def asRowMatrix(X):
     if len(X) == 0:
          return np.array([])
     mat = np.empty((0, X[1].size), dtype=X[1].dtype)
     for row in X:
          mat = np.vstack((mat, np.asarray(row).reshape(1,-1)))
     return mat
```

This function transforms an images matrix as a vector. It takes as input a matrix A and returns a vector mat.
The function vstack returns an vertical vector.

- Function for calculating the average and centering all images

```
def computeMeanImage(matImg):
    meanImage=np.mean(matImg, axis=0)
    return meanImage


def computeCentredImage(q, p):
    matImg=q-p
    return matImg
```

The function ComputeMeanImage takes as input the matrix matImg and returns the average image for the learning base.
The function ComputeCentredImage center each image .The average image is subtracted from each image
It returns back matIm:corresponding to the result of the image centered.

- The function for calculating the covariance matrix

```
def matrixCov(H): # la matrice de covariance
    matrice_cov=np.cov(H)
    return matrice_cov


def matrixCorr(A):
    matrice_corr = np.corrcoef( A,rowvar=0)
    return matrice_corr
```

corrcoef standardize the covariance matrix.

- Functions for calculating specific elements

```
57 def printEigenComponent(eigeinvalues, eigenvectors):
58     for idx, val in enumerate(eigeinvalues):
59         print 'Lambda{0} = {1:.3f}'.format(idx, val)
60         print 'b{0} : {1}.T\n'.format(idx, eigenvectors[:,idx])
61 
62 
63 def computeEigenComponent(S):   # calcule des composants pricipales
64     EigeinValues, EigenVectors = np.linalg.eig(S)
65     printEigenComponent(EigeinValues, EigenVectors)
66     return EigeinValues, EigenVectors
```

The function printEigenComponent print eigenvalues and eigenvectors.
The function computeEigenComponent calculates eigenvalues and eigenvectors. They are computed with the normalized covariance matrix.

- Main funtion gathering invocation of other computing functions

```
17 def pca():
18     print '##########################################'
19     print '###                                    ###'
20     print '###   Analyse en composant principale  ###'
21     print '###                                    ###'
22     print '##########################################'
23 
24     Subjects = np.array(['S '+str(num) for num in np.arange(1,99)])
25 
26     [X,y] = read_img("G:\\ProjetV1.0\dataBaseImages")
27     print np.array(X).shape
28     print y
29     return
30     matImg=asRowMatrix(X)
31     [w,z]=matImg.shape
32     print "la matrice des images  :\n', matImg
33     print 'les dimensions de matImg  :\n', [w,z]
34 
35     MU= computeMeanImage(matImg) # calcule de la moyenne
36 
37     CenImg=computeCentredImage(matImg, MU) # centrage des images par rapport a la moyenne
38 
39     print MU.shape
40     plt.figure()
41     plt.imshow(MU.reshape((112,92)), cmap='gray')
42     return
```

This function allows to call others functions and create projection axes of eigenvectors and we also saved the average image.