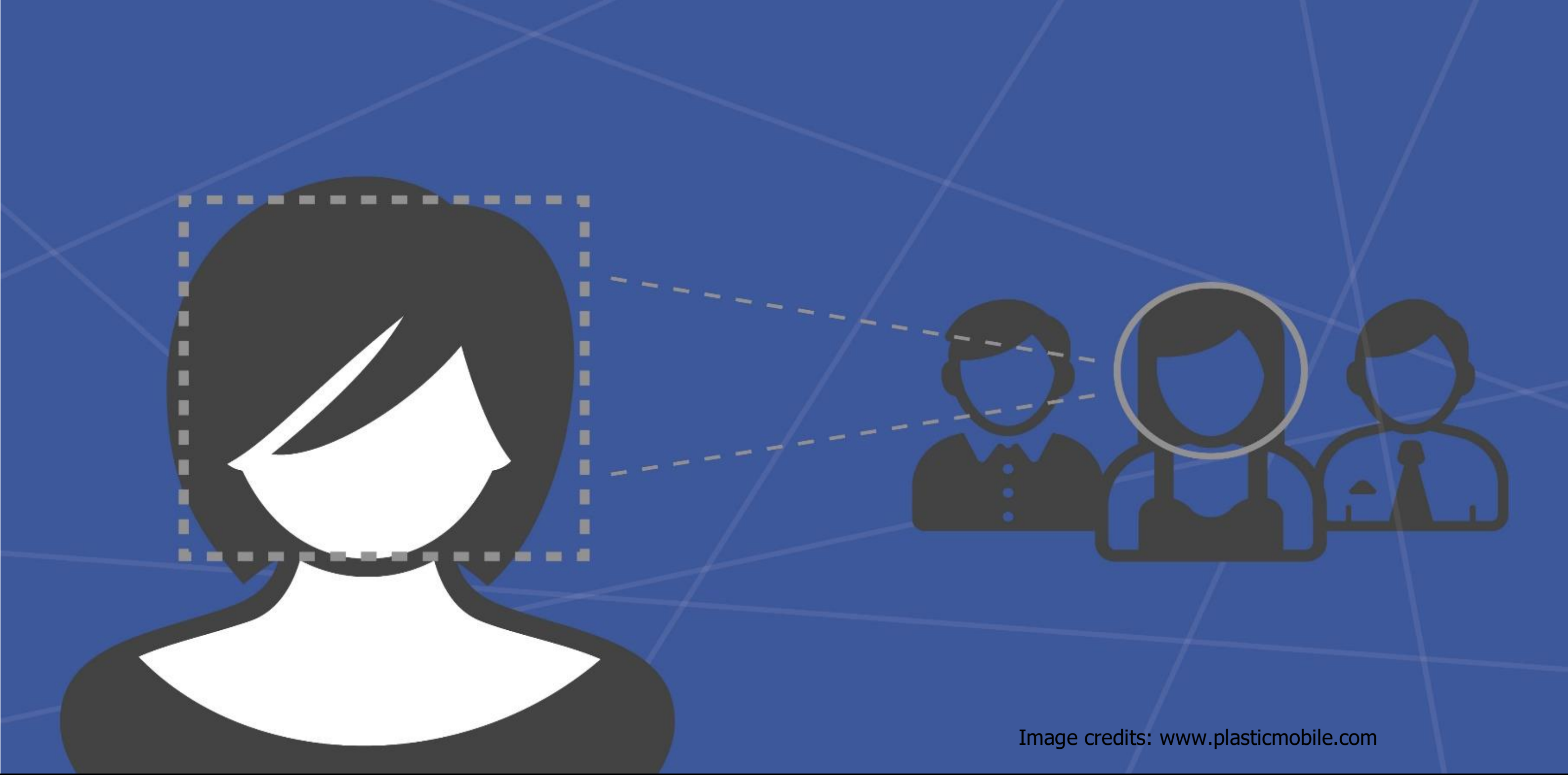


Image credits: www.plasticmobile.com

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

Development of a face recognition system using Subspace Methods

Internship report ST40 – A2015

DUONG Tony

Department of Computer Science

GI03

University of Tsukuba

COMPUTER VISION LABORATORY

1-1-1 Tennodai
305-8577 TSUKUBA, JAPAN
www.tsukuba.ac.jp

Tutor
FUKUI Kazuhiro

Suiveur UTBM
CAMINADA Alexandre

utbm
université de technologie
Belfort-Montbéliard

Introduction

Currently, I am completing an Engineer's degree in Computer Science at the Technological University of Belfort-Montbéliard. I am in my fourth year and therefore I have to conduct a 6-month internship.

I chose to do my internship abroad because I wanted to benefit from an international experience as well as being able to study a field I am interested in, computer vision. Computer vision is without doubt a field that will be used by everyone in a near future. I also had the wish to experiment a laboratory internship as pursuing my studies into research was one of my ideas post-graduating.

Furthermore, choosing Japan was not a choice made out of blue. I learned Japanese at the engineering school and I wanted to put in application what I learnt. Japan has always been a fascinating country to me, where technology and tradition mix together. Travelling there would make me experience so many new things.

I was certain that doing my internship there could only be beneficial to me.

In this internship report, I will describe my experiences during my internship period. This report contains a presentation of the university and especially of the laboratory I'm studying, the Computer Vision Laboratory. Then, I will explain the tasks and project done as well as some theory of the methods I used for my project, the Subspace Methods. The last part will summarize the results obtained and explain the future goals I have set after this internship.

Acknowledgements

I wish to thank all the people who have helped me throughout the internship:

- Professor **FUKUI Kazuhiro**, a head of the computer vision laboratory of Tsukuba University, for having accepted me as a member of this laboratory. I could have my first experience as a laboratory intern and I also learnt a lot of things related to Computer Vision thanks to him. His expertise and his advices helped me to accomplish my project.

- Professor **JIMBO Keiko**, a teacher of Japanese language at Universite de Technologie de Belfort-Montbeliard, who has helped me a lot for finding the best internship possible in a Japanese university. She put a lot of effort in it and I'm very grateful for that.

- Mr. **KOYAMA Takahiro**, a member of the computer vision laboratory and the tutor designated by Tsukuba university to help me after my arrival, who was always there to help me whenever I had problems related to paperwork. I really appreciate his welcoming attitude and I could accustom myself to the Japanese lifestyle quickly thanks to him.

- **All the members of the laboratory**. Thanks to their experiences, they could always help me and share their knowledge when I was in difficulty. I am very thankful for their friendly and helpful behaviour even when it was difficult to communicate due to the difference of language.

- Professor **CAMINADA Alexandre**, Mrs. **JACQUOT Mireille**, and **all the members of my home university UTBM** who made my internship to Japan possible. It was a dream for me to go to Japan and I spent unforgettable moments there thanks to you.

Contents

I. Presentation

A. University of Tsukuba

University of Tsukuba, one of the oldest national universities and one of the most comprehensive research universities in Japan, is located in the city of Tsukuba, Ibaraki Prefecture in the Kanto region of Japan. It is located in the north-east of Tokyo, and the Tsukuba express line connects these two cities, making it easier to travel between them.

Figure 1 – Location of Tsukuba in Japan

Strength

The academic strength comes especially from its STEMM fields (Science, Technology, Engineering, Mathematics, Medicine) and physical education, as well as related interdisciplinary fields by taking advantage of its location in Tsukuba Science City which has more than 300 research institutions. The research strength of Tsukuba is one of the reason that made me do my internship there.

Internationalization

Tsukuba is a university which is open to all within and outside of Japan. Indeed, as of August 2015, the university has over 300 international interuniversity agreements with universities from all around the world. I had the opportunity to not only meet Japanese people, but people from all around the world (England, China, Malaysia, Russia and more). This is another one of the strong points of Tsukuba University.

B. Computer Vision Laboratory

Computer Vision Laboratory is a university laboratory where all the members work on different aspects of computer vision. Many fields are studied: human sensing, robot vision, bio-image processing, but also theoretical research of pattern recognition (see Figure 2). The goal of the laboratory is to constantly improve the performance of the techniques used in the computer vision field.

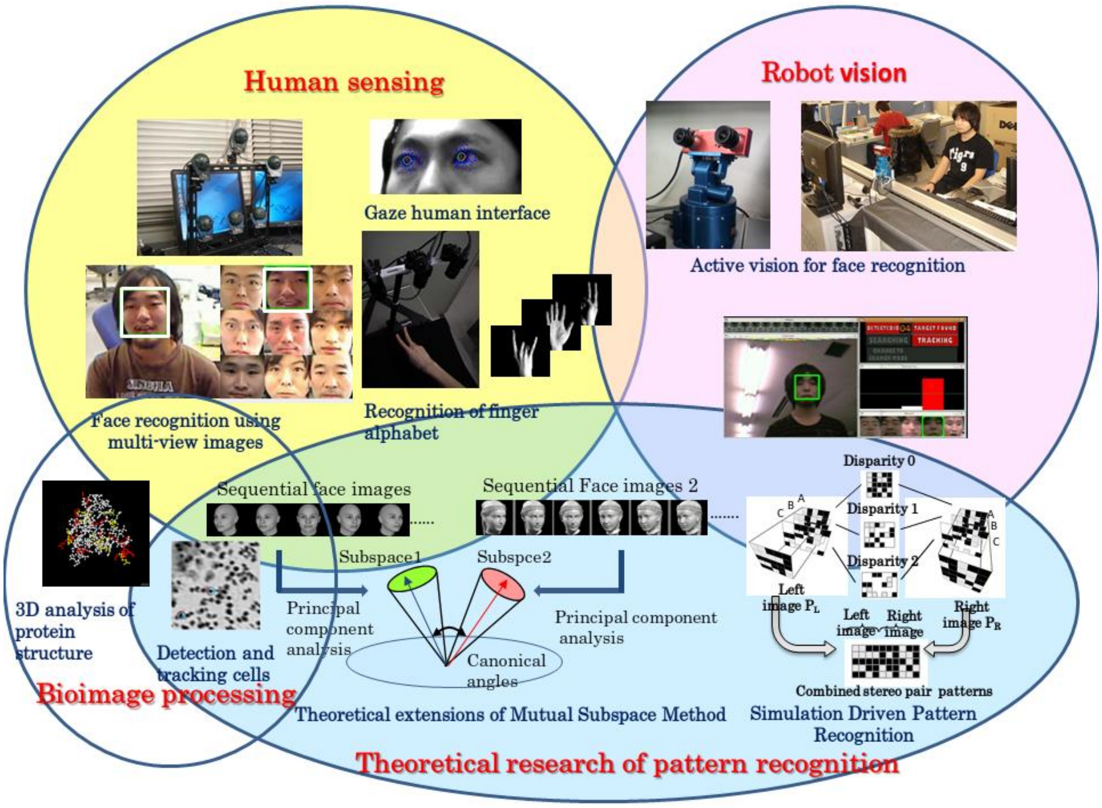


Figure 2 – Related works in the laboratory
Source : http://www.cvlab.cs.tsukuba.ac.jp/

The seventeen members are comprised of professors, doctor, master, undergraduate and exchange students as well as an administrative assistant. Fukui Kazuhiro, my tutor, is the head of the laboratory and the researches done in the laboratory by the students are for the most part revolved around his works. He has more than 25 years of experience in the field of computer vision and was a researcher at Toshiba, researching and developing various image recognition systems such as a face recognition system (FacePass). The students all have a research topic to conduct and sometimes participate in conferences, presenting their research work.

Almost half of the members come from foreign countries (Brazil, Indonesia, China, France) and the laboratory accepts regularly international students. All of them were very nice and helpful when I was stuck to a problem.

The environment work is typical of a computer science laboratory. Each member works on a desk with a two-screen computer and a camera for experiments as well as mine. Concerning the work methods, everyone conducts their own project and works autonomously. There is also a space in the laboratory for meetings or for free-time. Meetings are held once a week where the students do their presentations (research progress, preparation for the final presentation…).

The most notable contributions of the laboratory in the computer vision field is the development and the use of 'Subspace Methods'. These are used for character recognition, face recognition, object recognition, voice recognition, text classification and also data mining. Having experienced it through my project, I can tell this is a powerful method for classification. The theory shall be explained in the next part.

II. Project, Tasks and Activities

A. Project

Before starting my internship, the project topic has been defined: Development of a face recognition system using Subspace Methods.

As the name suggests, I was assigned to develop a face recognition system by using the methods that my tutor has developed for the past years, the *Subspace Methods*. I will explain the theory of these method later in the report.

The application has two main tasks:

- Detect faces from a web camera
- Recognize that face (identify the face)

This kind of application could be used in various situations.
For example, it could be useful for the development of automatic door access using face recognition and also for the field of augmented reality where one can recognize anybody just by watching them (this could lead to privacy problems but that is another problem).

Besides, this project seemed very interesting to me because face recognition is a topic known by everyone in the society even if it is not enough advanced for real application. Being able to learn more about it will certainly help me in the future as I intend to work in a field related to new technologies after graduation.

For this internship, basic knowledge of linear algebra and statistics was necessary since mathematics is omnipresent in the domain of computer vision. The courses taught in my home university were very helpful for doing the project.

The project has been done on the Matlab R2014b software.

B. Tasks

Before starting the development of the application, I had to study and read a lot of research papers about computer vision in general. It was difficult at first to know where and how to begin. After asking for advices from the members of the laboratory, some research papers and presentations were suggested to read first since I was a complete beginner in that field.

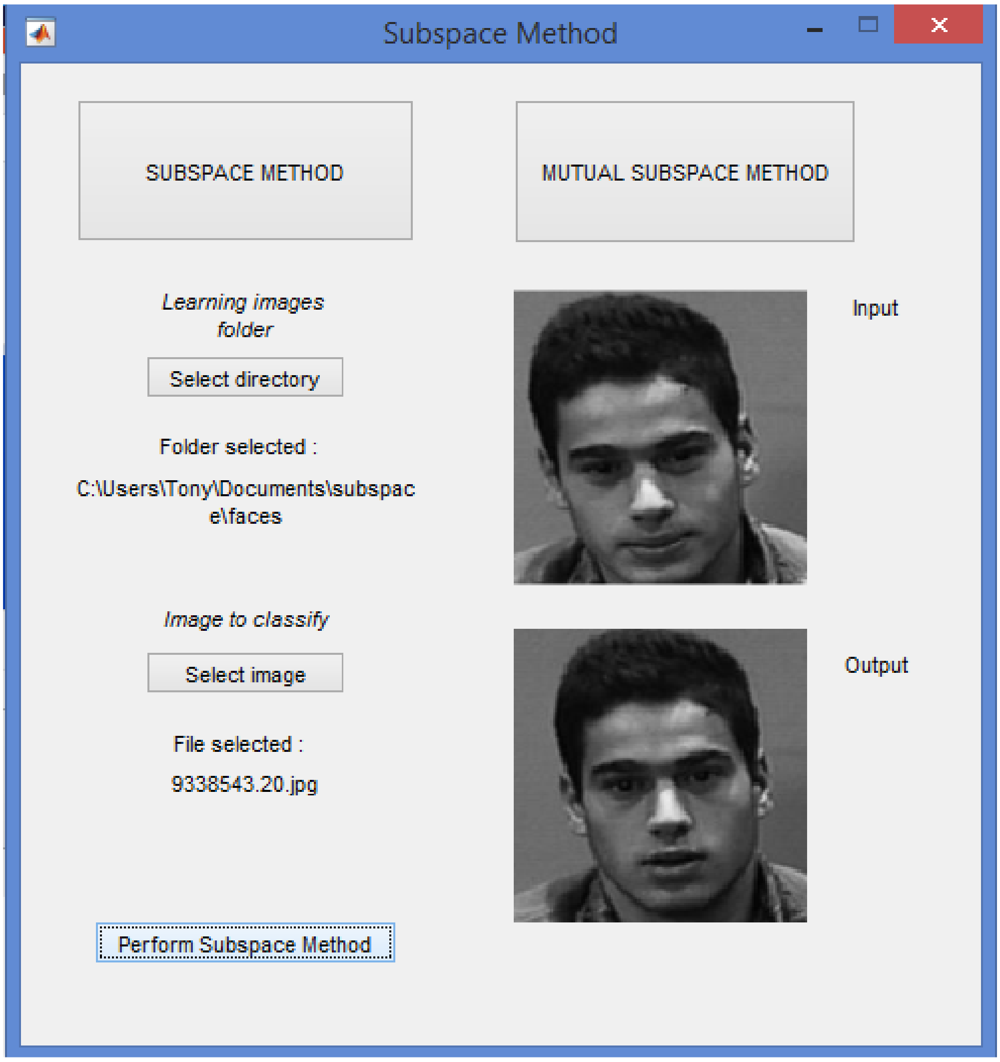
As shown in the next Figure, my main tasks on the preparation part were:
- Reading research papers
- Implementing the algorithms on Matlab of some papers
- Developing a minimalistic face-recognition application using these algorithms (see Figure 3)
- Analyze the results
- Studying mathematics again (subspaces)

Once a month, I also wrote a report that showed the progress of my studies to my tutor. In these reports, I detailed the methods or techniques learnt by reading papers or courses on the Internet. This ensures whether my understanding was correct or not. The subjects studied during the internship were from the start:
- Principal Component Analysis
- Linear Discriminant Analysis
- K-Nearest Neighbors Algorithm
- Subspace Method
- Mutual Subspace Method
- Constrained and Orthogonal Subspace Methods
- Kernel Subspace Methods
- Viola-Jones Face Detection Algorithm

Of course, most of these methods will be explained in a simple way in this report. I also read about many other methods and techniques but not all were useful for the development of the project.

Then, in the monthly report, I also described the application realized during the past month and showed the results of it. The first application was for instance a minimalistic face recognition system covering the subspace method and the Principal Component Analysis. The difficulties met during my work were also described, so Professor Fukui could help me.

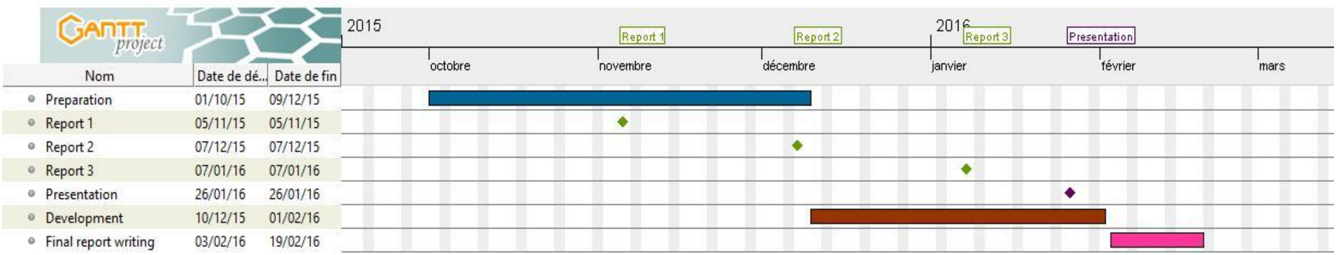
Figure 3 - Minimalist application for algorithm testing



After studying the methods related to face-recognition and face-detection, I started the development of application of the project.

You can see the overall planning of the internship in the Figure 4.

Figure 4 - Planning of the internship



III. Theory and experiments

A. Principal Component Analysis (PCA)

Learning this method was the first "right" step in my internship. Indeed, according to a colleague researcher, if I could understand the Principal Component Analysis, this was going to help me a lot for reading papers related to computer vision.

At first, I tried to dive in directly by reading several research papers of my professor Fukui Kazuhiro. But I was not able to Figure out some of the techniques used by my professor. Then, after learning about Principal Component Analysis, a lot of things became much clearer.

The Principal Component Analysis (or PCA) method is used a lot in computer vision algorithms. Indeed, it is a linear transformation technique used to emphasize variation in a dataset and bring out strong patterns in a dataset. It uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called **principal components**.

Let's consider the following dataset as shown in Figure 5. The data of this dataset seems correlated as it looks like that when x grow, y grows too and this in a linear way.

This example applies PCA in a two-dimension space but PCA can and will likely be used in higher dimension space.

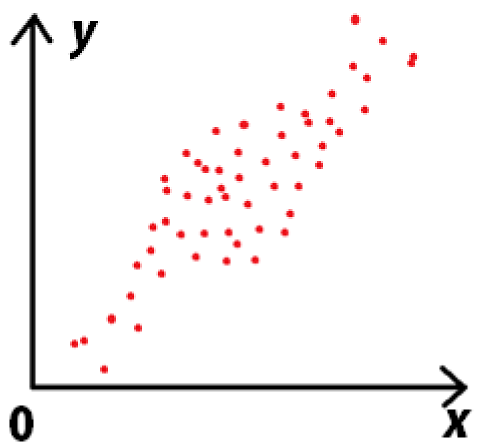


Figure 5 - Distribution of data in two-dimensional space

In this case, PCA will bring out a one-dimensional subspace, represented by a green line in Figure 6 that would most likely represent the variation of this dataset. In other words, this would be a line where the projection of the dots onto this line will be as sparse as possible.

We would get a line similar to the one in the next Figure. This is the first principal component.

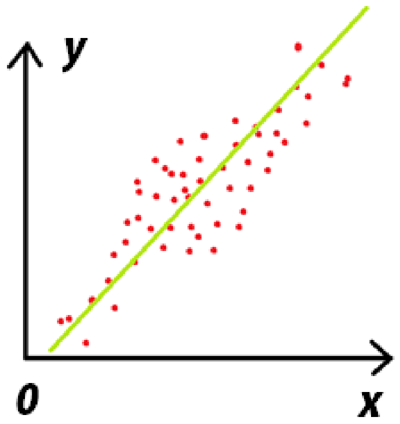


Figure 6 - First principal component

The number of principal components is less than or equal to the number of original variables which are in this case the coordinates x and y, that being 2. As said before, the first principal component has the largest possible variance. Each succeeding component (in this case, only one is left) has the highest variance possible under the constraint that it is orthogonal to the preceding components.

Considering this assumption, the second and last principal component would be a line similar to the orange one in the next Figure.

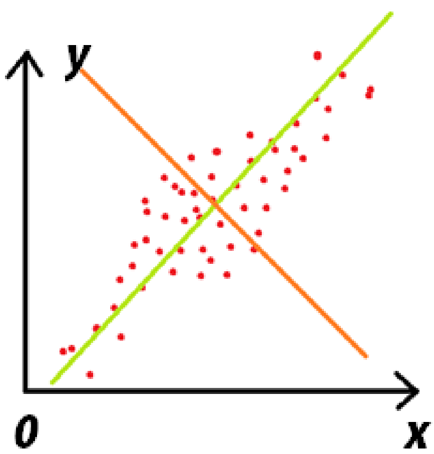


Figure 7 - Second principal component (orthogonal to the first one)

Why would PCA be useful for computer vision?

In the case of computer vision, PCA is very useful for its **dimensionality reduction** ability. Indeed, in the previous example, we could have reduced the dimensionality from two to one and change the basis from the original basis (in black) to just a one-dimension basis (green axe).

Figure 8 - Projected data onto the first principal component axis

Of course, a part of the information may be lost but for the sake of shortening the computation time, this process is worth dropping the other dimension.

In subspace methods, an image is represented by a vector which is obtained by concatenating all the rows of an image matrix as a matrix of the size of the image (height*width) in which each element is an integer ranged between 0 and 255 for a grayscale image. For example, an image with the size of m*n can be represented by a vector in m*n dimensional vector space, as shown in Figure 9.

As a result, the dimensionality would be equal to the number of pixels in an image. Even for a small image of 100*100 pixels, the dimensionality would be equal to 10000, which is too big for computation.

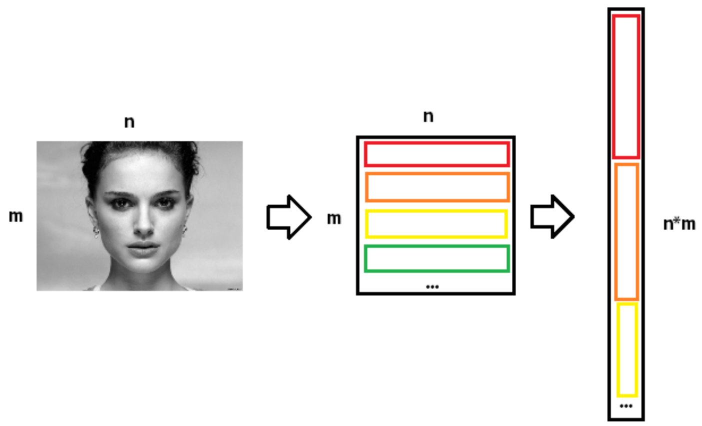


Figure 9 - Vectorizing of original grayscale image to feature vector

In the case of face recognition and subspace methods, all image vectors from a same person (or class) are grouped by concatenating them into a matrix.

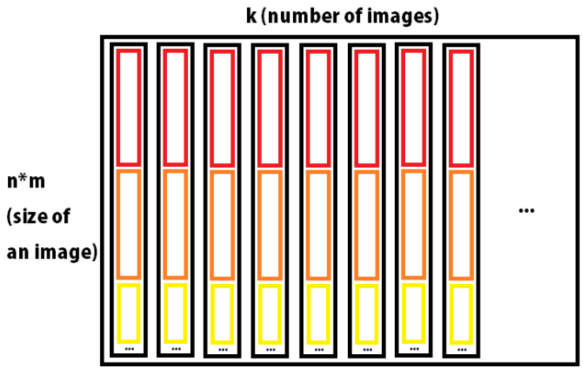


Figure 10 – Feature matrix obtained by concatenating feature vectors

Finally, the Principal Component Analysis is applied to get the best principal components. The obtained principal components shall constitute a subspace that represent the original data.

This was a long explanation about the PCA method. However, since it is at the base of the subspace methods, I considered that it was important to elaborate it clearly.

B. Face recognition : Subspace methods

My project is the development of a face recognition application so I should look for the best possible technique for recognizing faces. My professor suggested me to study the subspace methods which are widely used in the field of computer vision. He has worked a lot on that theme for the past years, so my learning resources mainly come from his research papers.

Subspace methods are classification methods by comparing and classifying subspaces. Last year, I learned about the concept of subspaces through the classes taken at UTBM (MT45 and MT42) and they were really useful for the comprehension of these methods.

In the previous part, we learnt how to generate subspaces from a set of images. Now, the next problem is to measure the similarity between two subspaces for a classification task.

Face recognition consists of two phases:
- Training
- Recognition

Training

In order for the "system" to learn how to classify new input sets of data, it has to be trained first. Training the system is an indispensable task to do. We give to the system several sets of data so that it can identify them.

Recognition

After the system has been trained, recognizing a new input face becomes possible. We can then ask the system to classify a new set of images based on what he knows already from the training part.

There are many extensions of subspace methods. Here are the examples:
- Subspace Method and its advanced **Constrained** and **Orthogonal** versions
- Mutual Subspace Method and its advanced **Constrained** and **Orthogonal** versions

Subspace Method is used for comparing a vector with a subspace (only one image of the face to classify).

On the other hand, Mutual Subspace Method is used for comparing two subspaces (many images of the face to classify).

Each one of them has their respective **kernel version**, which is used in case when the data are not distributed linearly (this will be explained later).

FRAMEWORK

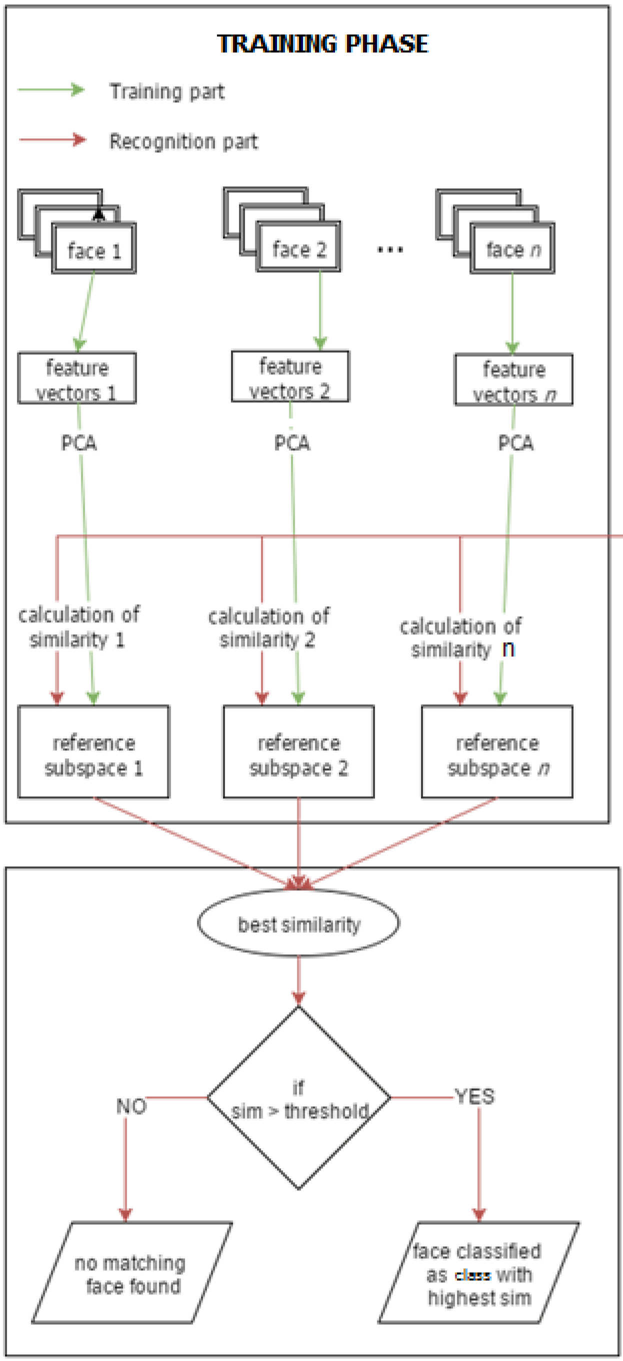


Figure 11 - General framework of the mutual subspace method

How to calculate the similarity between two subspaces?

The similarity between two subspaces is a numerical value that indicates how much these subspaces look like each other.

This calculation of the similarity is common in all the versions of the subspace methods.

The similarity between two subspaces is defined based on the canonical angles between them.

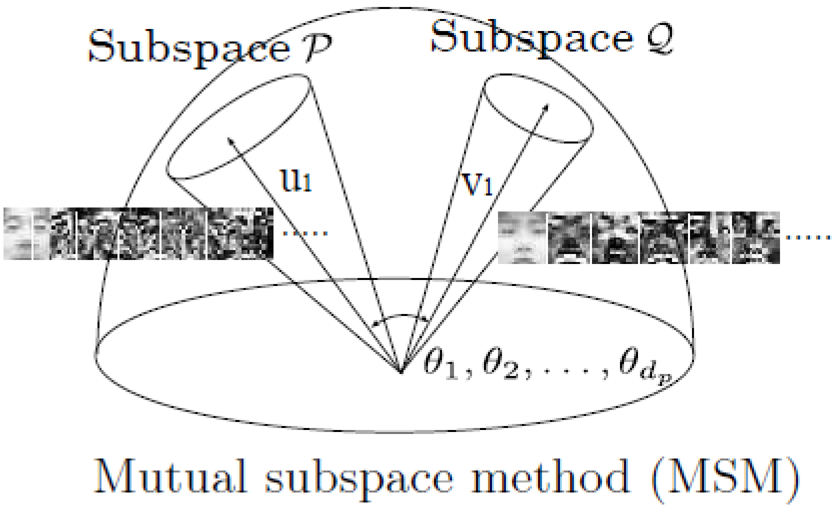


Figure 12 - Canonical angles (or principal angles) between subspaces

Given two subspaces *P* and *Q* of the same dimension *n*, the canonical angles are uniquely defined as:

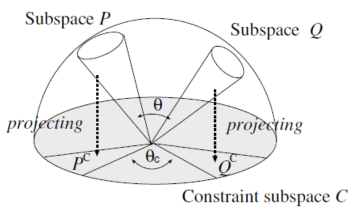
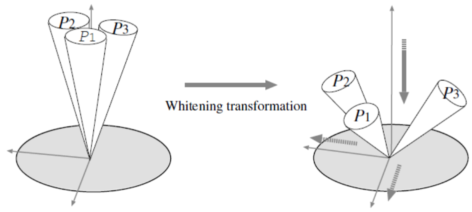
$$\cos^2 \theta_i = \max \frac{(u_i \cdot v_i)^2}{\left||u_i|\right|^2 \left||v_i|\right|^2}$$

where $u_i \in P$, $v_i \in Q$, $\left||u_i|\right|$ and $\left||v_i|\right| \neq 0$, $(\cdot)$ and $||\quad||$ represent an inner product and a norm respectively.

The advanced subspace methods: Constrained and Orthogonal Mutual Subspace Method

I will now describe through Table 1 what are the characteristics of the advanced version of Mutual Subspace Methods.

Table 1 - Description of CMSM and OMSM

Constrained Mutual Subspace Method	Orthogonal Mutual Subspace Method
A constrained subspace *C* (the details about how to generate a constrained subspace will not be explained in the report) is generated beforehand. Then, we project both reference subspaces and input subspace onto *C* and calculate the similarity between the projected subspaces.  	A whitening matrix (the details about how to calculate a whitening matrix will not be explained in the report) is calculated beforehand. Then, we apply this transformation to both reference and input subspace. Finally, we calculate the similarity. 

Both methods have more classification power than the basic Mutual Subspace Methods because of the feature extraction step added in these two methods. Learning these methods required me to learn the detailed algorithm of Mutual Subspace Method in advance.

To put in application what I learnt about Subspace Methods until that time, I implemented from scratch each one of these methods and I developed an application on MATLAB using them for facial recognition. The application shown in Figure 13 displays the accuracy of the method chosen by giving it a folder containing the images for training and another folder containing the images for testing.

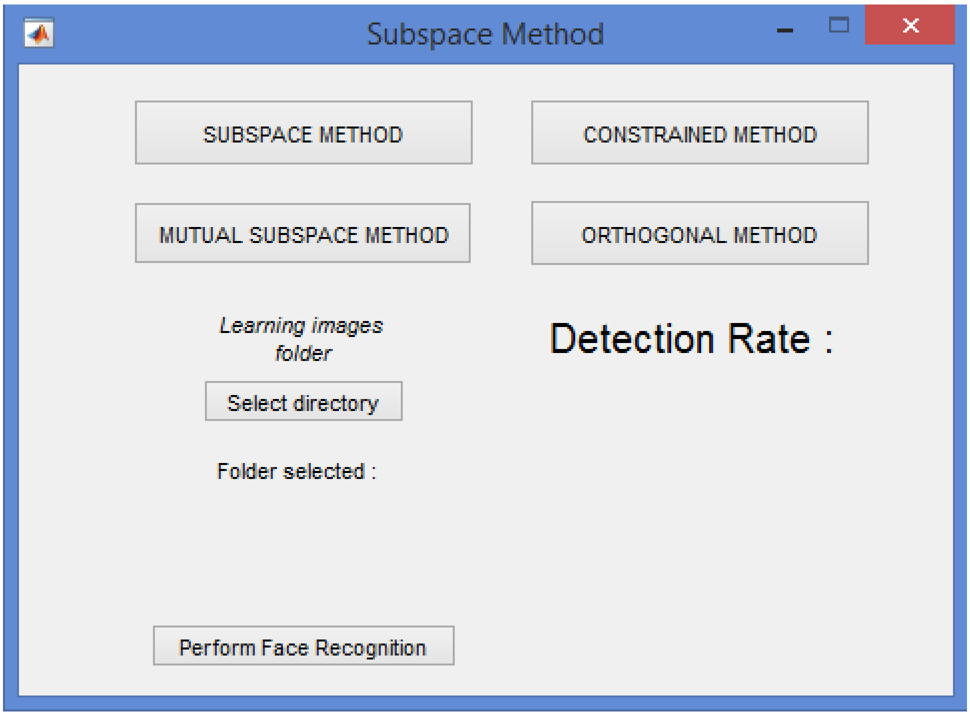


Figure 13 - Application calculating detection rate

How to evaluate the efficiency?

The idea is to divide a set of face images of each person into two groups. One group will be used for training and the other one will be used for testing as shown in Figure 14.

In other words, the system will be trained first with the training data. It will learn the faces of each person based only on the training data.

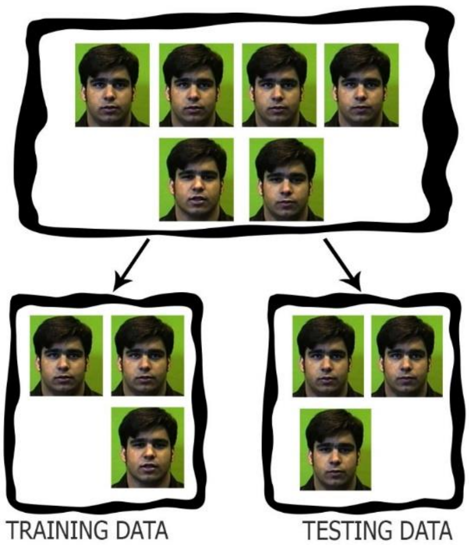


Figure 14 - Separation of a set of face images into training group and testing group

After training the system, we then proceed to evaluate the performance by using the testing data.

Finally, after comparing all the training data and the testing data of each subject, we get a similarity matrix S of size $n * n$ where n is the number of subjects.
The element S_{ij} of the matrix is the similarity between the testing data of class i and the training data of class j $(0 \leq i, j \leq n)$

$$\begin{array}{cc} & \text{Training data} \\ \text{Testing data} & \begin{bmatrix} S_{11} & S_{1j} & & S_{1n} \\ \vdots & \ddots & \cdot\cdot & \vdots \\ S_{i1} & Sij & & S_{in} \\ & & & \\ S_{n1} & S_{nj} & & S_{nn} \end{bmatrix} \end{array}$$

<u>First experiment</u>

For this first experiment, I used a dataset called "faces94" composed of only frontal images of 113 male and 20 female subjects.

Following are the database characteristics.

- Image resolution: 180*200 pixels
- Number of images per person: 20
- Background: plain green
- Position of face in image: minor changes
- Head turn, tilt and slant: minor changes
- Image lighting variation: none
- Expression variation: considerable expression changes

Figure 15 - Samples of the faces94 database

Now, I would like to evaluate the accuracy of the different subspace methods using this database.

In the first experiment, I separated the 20 images of each person into one group of 10 images for training and another group of 10 images for testing.

	MSM	CMSM	OMSM
Recognition rate	100%	100%	100%

We could obtain perfect recognition rates. This is due to the fact that the dataset contains only well segmented front faces. We can deduct that for front face recognition task, the subspace methods work effectively.

Second experiment

This time, I used the CMU-PIE dataset which is more complex due to the lighting variation.

Following are the database characteristics.

- Image resolution: 32*32 pixels
- Number of images per person: 80
- Background: none
- Position of face in image: always centered
- Head turn, tilt and slant: none
- Image lighting variation: yes
- Expression variation: none

Figure 16 - Samples of the CMU-PIE database

In this experiment, I kept 20 images for training and used the rest of 60 images for testing.

Here are the results.

	MSM	CMSM	OMSM
Recognition rate	87.5%	94.53%	91.41%

The recognition rates still remain high although they are not perfect.

We can see that Subspace Methods could potentially be a very suitable algorithm for face recognition. Nevertheless, more advanced versions of these algorithms have been developed. In more complex situations than the first two experiments, these advanced methods may be more suitable.

Kernel Subspace Methods and Kernel PCA

In face recognition, the normal versions of the subspace methods work fine with "easy to classify" sets of face images, such as in Figure 17. That is to say, set of images in which illumination, face angles do not vary a lot and environment does not change.

Figure 17 - Example of "easy to classify" image
Source: faces94 database

But in real application, it is hard to get such a dataset. Angle of face and direction of lightsource always vary a little bit and pictures are not always taken under the same environment as in Figure 18.

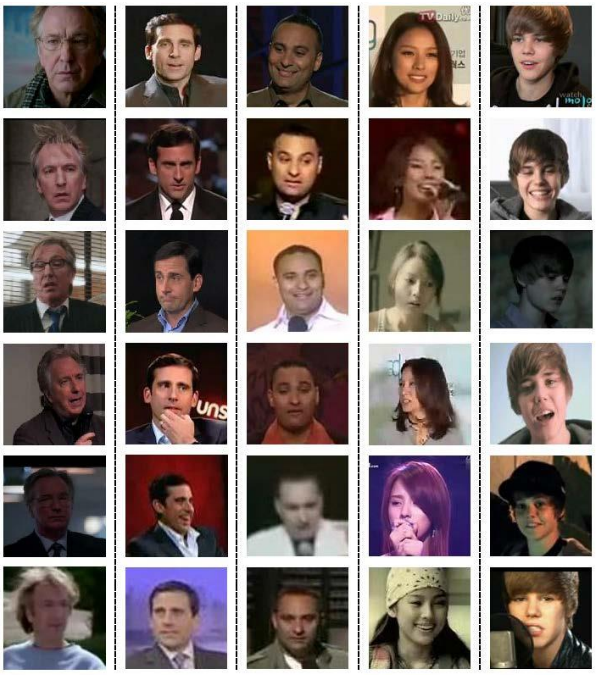


Figure 18 – Example of hard dataset
Source: MSRA-CFW database

For these kinds of images that are harder to classify, the most effective way is to use the kernel methods that are able to handle non-linear distribution of data. This non-linearity is caused by variations of the illumination, face angles or face expression in the case of face recognition.

Theoretically speaking, in the Figure below, using simple PCA would not be efficient at all if I want to bring out the strong patterns of the original data. We could not be able to separate the data in the best way with only a line.

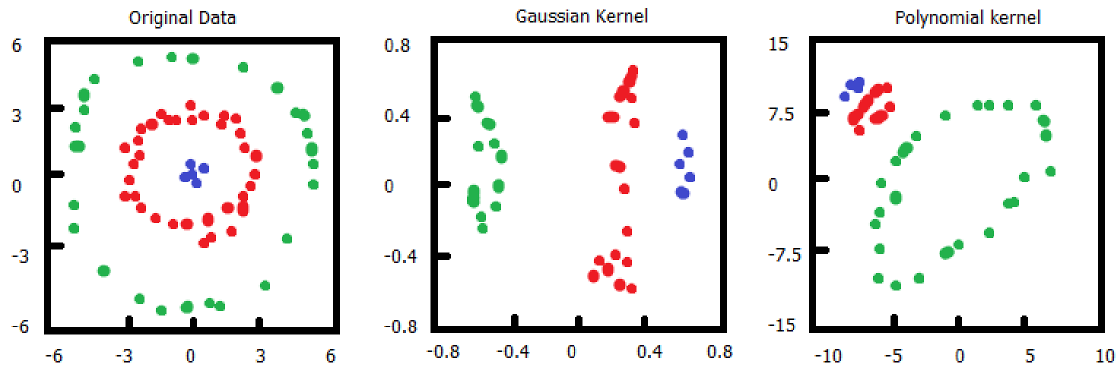


Figure 19 – Non-linear distribution and application of Kernel Principal Component Analysis

That is where Kernel PCA become useful because they can separate such data (see Figure 15). In Kernel Subspace Methods, we use the subspaces generated by Kernel PCA instead of those generated by basic PCA. Then, we can proceed to transformation such as whitening the data in the case of Orthogonal MSM or to the generation of a Constrained Subspace in the case of Constrained MSM.

How to get these principal components?

In theory, the idea is to map the original data to a higher dimensional space (Figure 20) by a transformation Φ. So the original data of dimension d_o, $(x_1, x_2, \dots, x_n)$ become $(\Phi(x_1), \Phi(x_2), \dots, \Phi(x_n))$.

Then, we apply the algorithm of PCA with these mapped values.

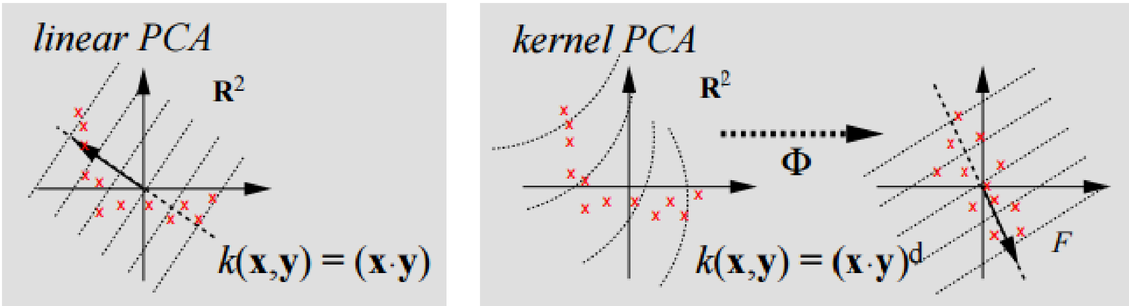


Figure 20 - Linear PCA, Kernel PCA and the mapping of the original data
Source: Kernel Principal Component Analysis by Bernhard Scholkopf, Alexander Smola, Klaus-Robert Muller

But in practice, it may not possible to find explicitly this higher dimensional space because the downside of this direct approach is that the computation becomes increasingly expensive.

I will not explain the details about the trick used to resolve this problem but the main idea has been explained. Even though I mentioned that Kernel PCA handles non-linear data better that Basic PCA, it can also handle linear data too.

An experiment will be presented after the face detection explanation.

C. Face Detection: Viola-Jones

After having explained the techniques used for recognition, I will now explain the method used for detecting a face in an image. The Viola-Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones. Although it can be trained to detect a variety of object classes such as animals or objects, it was motivated primarily by the problem of face detection.

I chose this framework because I needed a face detection framework that can be run in real-time with a correct detection rates. Indeed, the Viola-Jones algorithm has an extremely fast feature computation and has been already used for various face recognition.

How does it work?

The basic idea is to slide a window across the entire image and evaluate if a face is in that window at every location by applying rectangle filters (Figure 21). These rectangle filters are called Haar features.

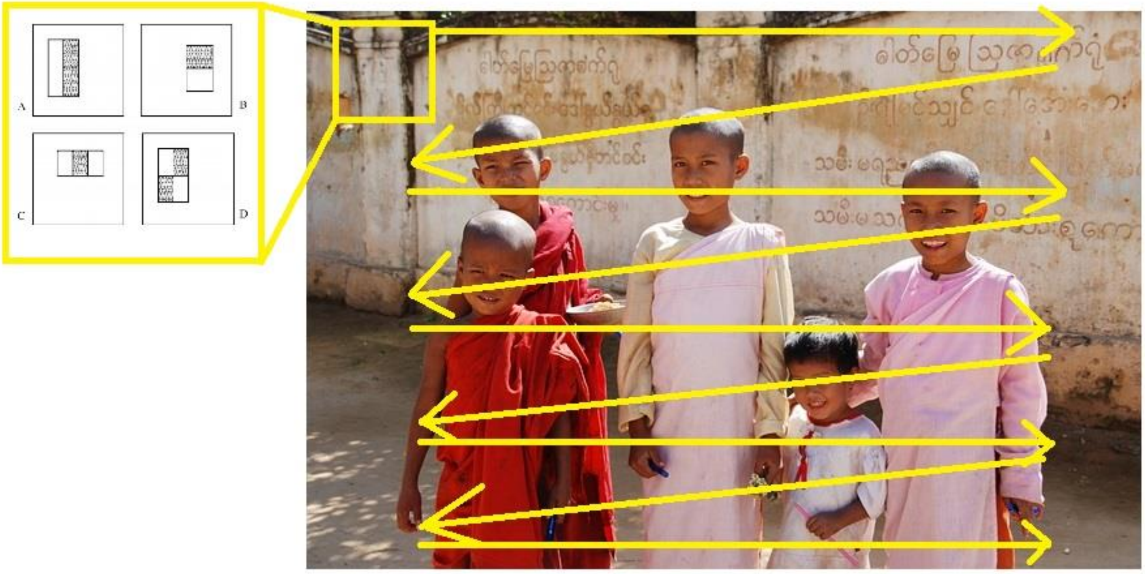


Figure 21 - Viola-Jones

Haar features

There are four types of Haar features (the ones in Figure 21). But with only these four types of features, in a 24*24 pixels input image, around 160 000 distinct features can be generated by modifying the scale, position and form in all possible ways under the condition that it is contained inside the image. You can visualize some of them in Figure 22.

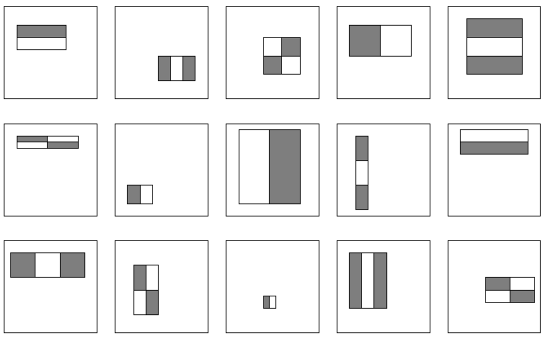


Figure 22 - Different Haar features

Nevertheless, most of these features are irrelevant. The best features selection is done by the Adaboost algorithm.

In order to train the system, we give it a lot of face images and non-face images. From them, Adaboost algorithm drops all the Haar features useless for face detection and only keeps the best features that represent best the face features.

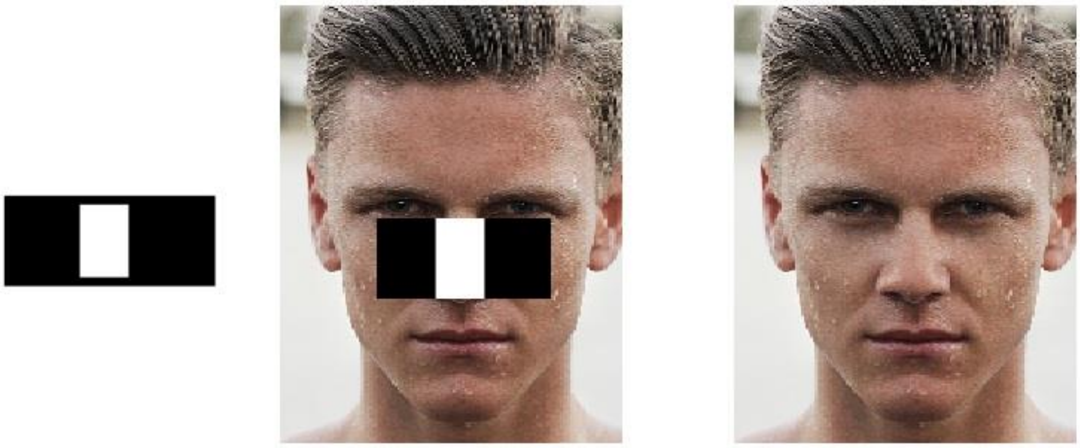


Figure 23 - Haar feature that looks similar to the bridge of the nose

In the Figure 23, this Haar feature will be kept because it represents the best the nose bridge of faces. Indeed, in general grayscale face pictures, the nose bridge will always be "whiter" than the sides of the nose.

Cascade classifier

Then, a cascade classifier as in Figure 24 is used.

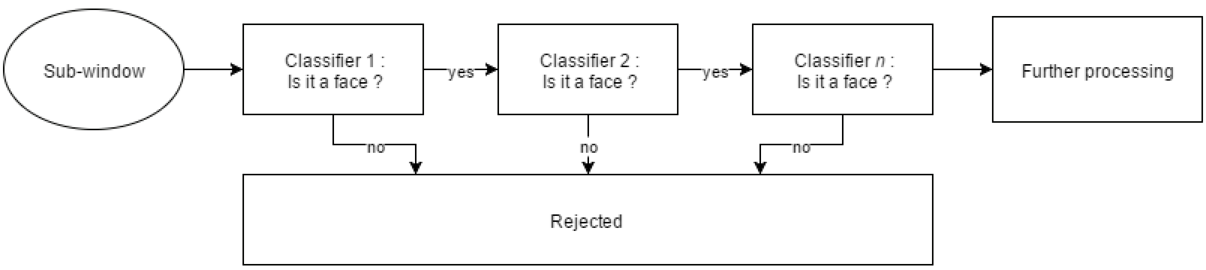


Figure 24 - Cascade classifier

For each sub-window of the original image, we process it through this classifier. Each stage of the classifier has the chosen Haar-features from Adaboost.

When a sub-window reaches a stage, the classifier will apply the Haar-features and this sub-window will get a value indicating its "face similarity". If this value exceeds the threshold of the stage, then it will go to the next stage.

The training of the classifier has not been done by me. For my application and for experiments, I took from the OpenCV library one classifier already trained. Indeed, a good training of the classifier requires lots of face images and non-face images and according to the research papers, such a training would last several weeks.

That is why I decided to not do the training but just focusing on the detection algorithm part.

The trained classifier (which is a xml file) I chose for my application is the one from the OpenCV GitHub repository.

Experiment - Face Detection only

For this experiment, I chose a database called MSRA-CFW from the Microsoft Research webpage composed of only random images of celebrities. It contains 202792 faces of 1583 people. For this experiment, I only selected 20 faces of 10 people. The classifier of my experiment is suited to detect frontal faces only so I chose images from the dataset according to that criteria.

The result I want of this experiment is the detection rate for such a database.
In this dataset, images are very different from each other in the sense that people can have different face expressions, face angles and illumination also vary considerably. In addition, characteristics such as

Figure 25 - Some of the original images of the MSRA-CFW database

hair, the wearing of glasses can differ from one image to another for a same person.

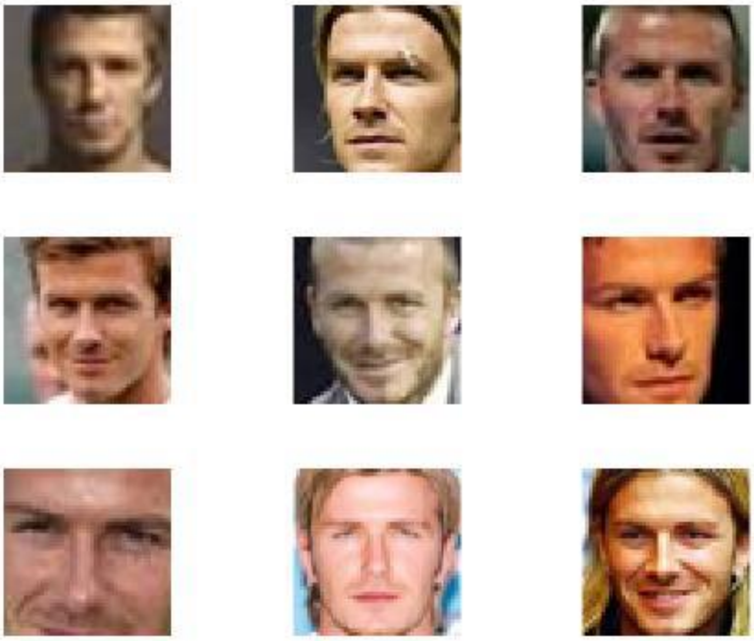


Figure 26 - Samples of the cropped images

The results for each subject can be found in Table 2.

Table 2 - Detection Rates

	Face detected	False face detected	Face non detected	Total images	Detection Rate
Adam Sandler	17	1	2	20	85
Alex Ferguson	8	5	7	20	40
Barack Obama	13	1	6	20	65
Brad Pitt	18	1	1	20	90
Cristiano Ronaldo	19	1	0	20	95
Julianne Moore	16	0	4	20	80
Kelly Hu	12	2	6	20	60
Leona Lewis	17	1	2	20	85
Monica Belluci	13	1	6	20	65
Reese Witherspoon	16	1	3	20	80

The results are quite favorable. The dataset was very complex and Viola-Jones algorithm could manage to detect correct faces in most of the images with few false positive apart from the subject Alex Ferguson. Indeed, this subject wear glasses in some of the images and this led to confuse the system.

Nevertheless, false face detected and face non detected are not an important problem for the application I intend to build. In fact, since my application will be used in real-time, the user could still adjust the position and the angle of his face if the face is not detected.

Experiment - Hard Dataset with Kernel Methods

In this experiment, I wish to test the efficiency of all the methods and show you that the kernel methods are the most suited for datasets with a lot of variations.

The same database as the one in the last experiment is used here: MSRA-CFW database.

For this case, I selected 10 classes (or people) and 40 images per class. Before proceeding to the training-testing part, I applied face detection on each of the image to extract only the face area of the images.

Figure 27 - Samples of the images after face extraction

Following are the database characteristics.

- Image resolution: 30*30 pixels
- Number of images per person: 40
- Background: varies
- Position of face in image: centered or slightly displaced to the side
- Head turn, tilt and slant: yes
- Image lighting variation: yes
- Expression variation: yes

Three sub-experiments have been done:

1) 20 images for training and 20 images for testing
2) 20 images for training, and two sets of 10 images for testing
3) 20 images for training, and four sets of 5 images for testing

The results of this experiment are shown in Table 3.

Table 3 - Accuracy of the subspace methods

Method	MSM	OMSM	CMSM	KMSM	KOMSM
Recognition Rate 1)	40%	50%	10%	50%	60%
Recognition Rate 2)	55%	55%	55%	75%	70%
Recognition Rate 3)	32.5%	15%	45%	47.5%	55%

We can deduce that the Kernel methods seem superior to their linear counterparts for such a database.

In order to choose the best methods possible for both detection and recognition, doing these experiments to check their efficiency has been very helpful.

According to the results, I decided to integrate Viola-Jones for the detection part and Kernel Orthogonal Mutual Subspace Method for the recognition part.

IV. Project

My project was "Development of a face recognition system using Subspace Methods". After studying the algorithms behind face recognition and face detection, I could use them for my application.

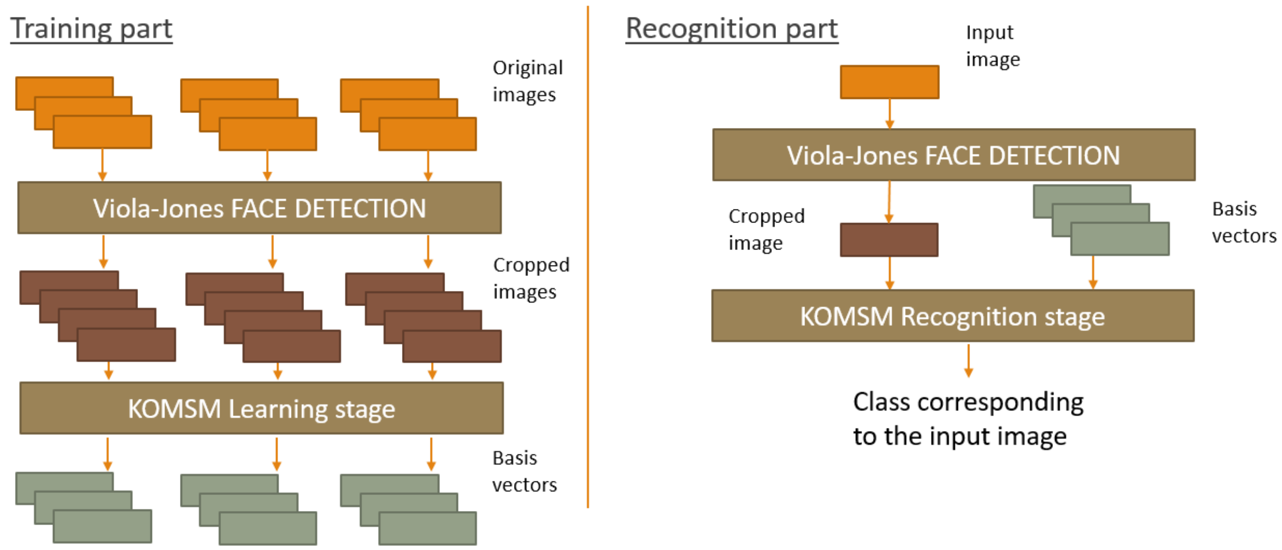


Figure 28 - Framework of the application

The face recognition application should be able to <u>recognize a face from the camera</u>. In other words, it should identify a face by displaying its name. But before that recognition task, it has to detect the face from the frame (picture taken from the camera).

The application is divided into two parts:

- Training part
- Recognition part

The general framework of the application can be visualized in Figure 28.

<u>Training part</u>

Before being able to recognize faces, the system must be trained by giving it face images with their name.

For that training part, I used Viola-Jones for detecting faces in the original images. The original images can either come from the camera or from images stored in the computer as shown in Figure 29.

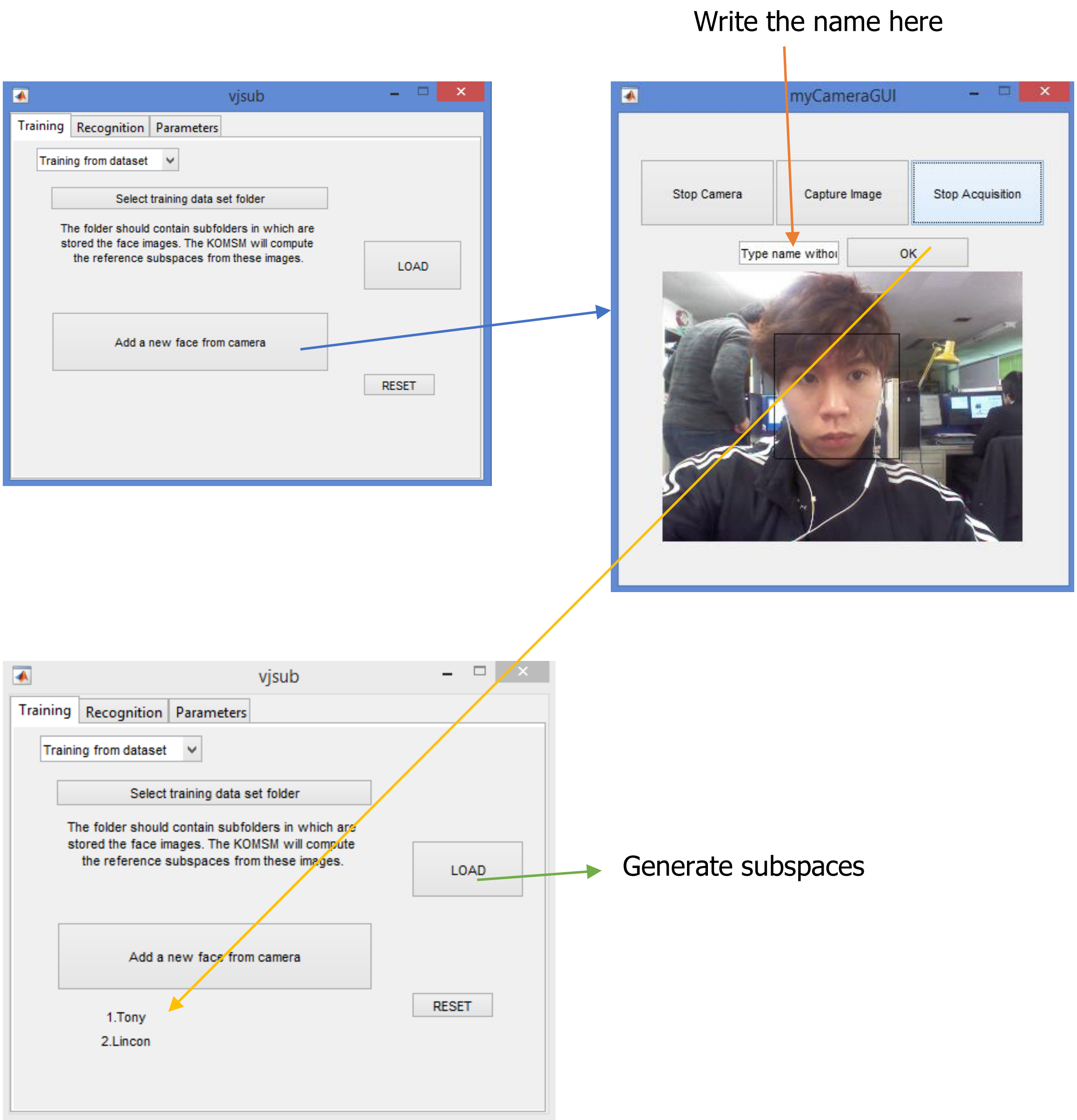


Figure 29 - Flow of the training part

Then, for each image loaded, after having detected the face, the image is cropped and only the face part of the image is kept (see previous section Face Detection: Viola-Jones).

Finally, these cropped images will be used for the training of the system. By using Kernel Orthogonal Mutual Subspace Method, reference subspaces that represents each face are generated.

<u>Recognition part</u>

Now, with these subspaces, the system can recognize faces from random images if the face is known by it.

In the system, original input images can either come from the camera or from images stored in the computer like in the training part.

Viola-Jones face detection will be applied first to the input image and after extracting the face image, it will go through the KOMSM algorithm. A black rectangle box will frame the face found in the images

Then, the image will be classified according to the similarity values resulted from KOMSM and the name of the person recognized will be written at the bottom of the application window.

The flow of the recognition is shown in Figure 30.

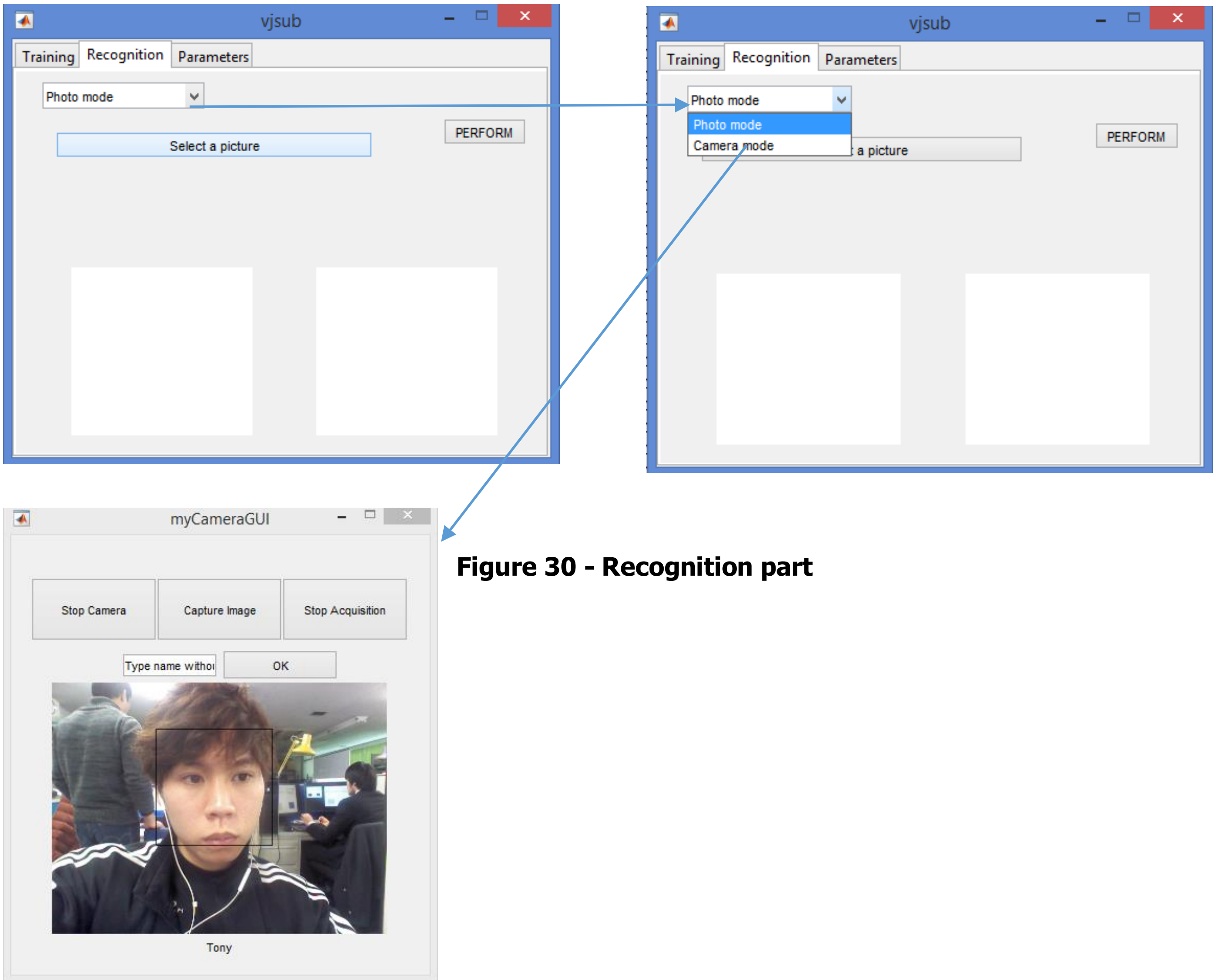


Figure 30 - Recognition part

By using the camera for real-time detection and recognition, this process will be done for each frame captured by it.

Parameters

Besides, it is possible to modify the variables as we please.

The variables that can be modified are:
- the dimensionality of the subspaces generated by PCA or kernel PCA.
- The kernel parameter (sigma) which is a parameter of a kernel function used in the kernel PCA algorithm.
- The size of the images (width and height) after being cropped by the face detection algorithm.
- The number of pictures taken by the camera for building the training subspaces
- The limit scale which is the minimum scale possible for a sub-window of the Viola-Jones (the scale starts from the size of the image and decreases until the sub-window reaches a size of 20*20 pixels).
- The maximum number of faces to detect in an image
- The timer period, which is necessary for the timer function used for real-time capture
- The number of frames to process from the camera

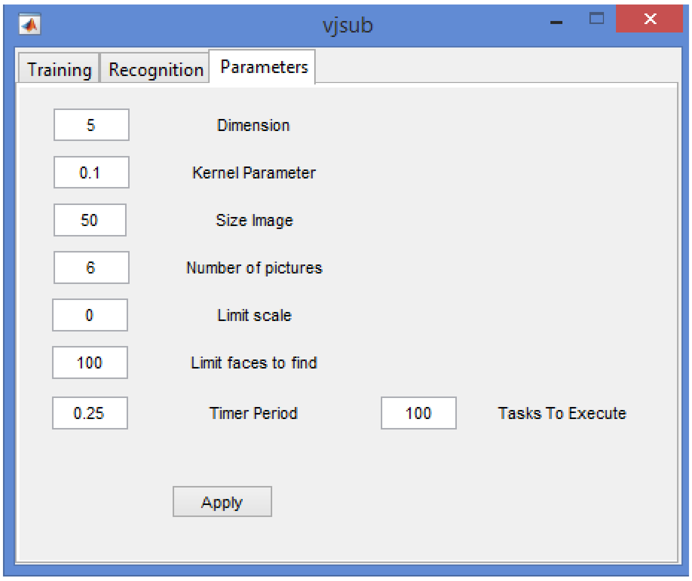


Figure 31 - Parameters

Results

I tested the performance of the application. From a functional point of view, the application works fine.

The application has no difficulty in finding faces from images or camera even with a complex background.

Besides, as you can see in the next graph, it is able to recognize quite accurately faces. When being in front of the camera, the system is capable of recognizing the person most of the time even when moving and tilting the face.

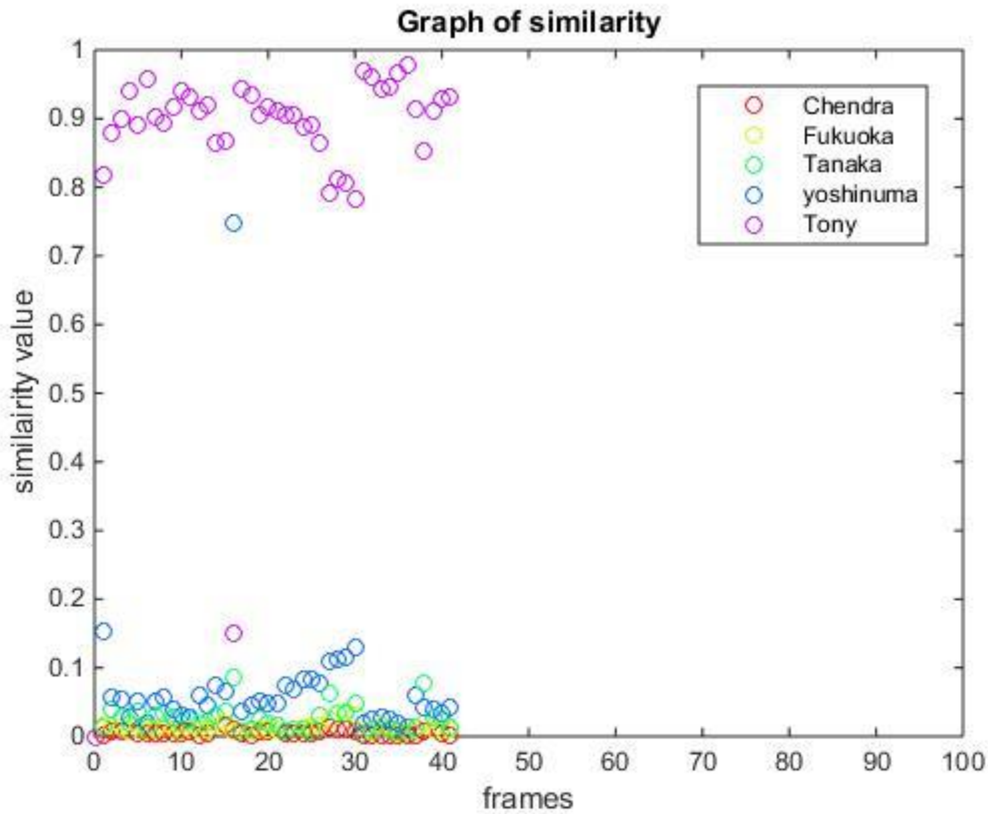


Figure 32 - Testing the camera with my face

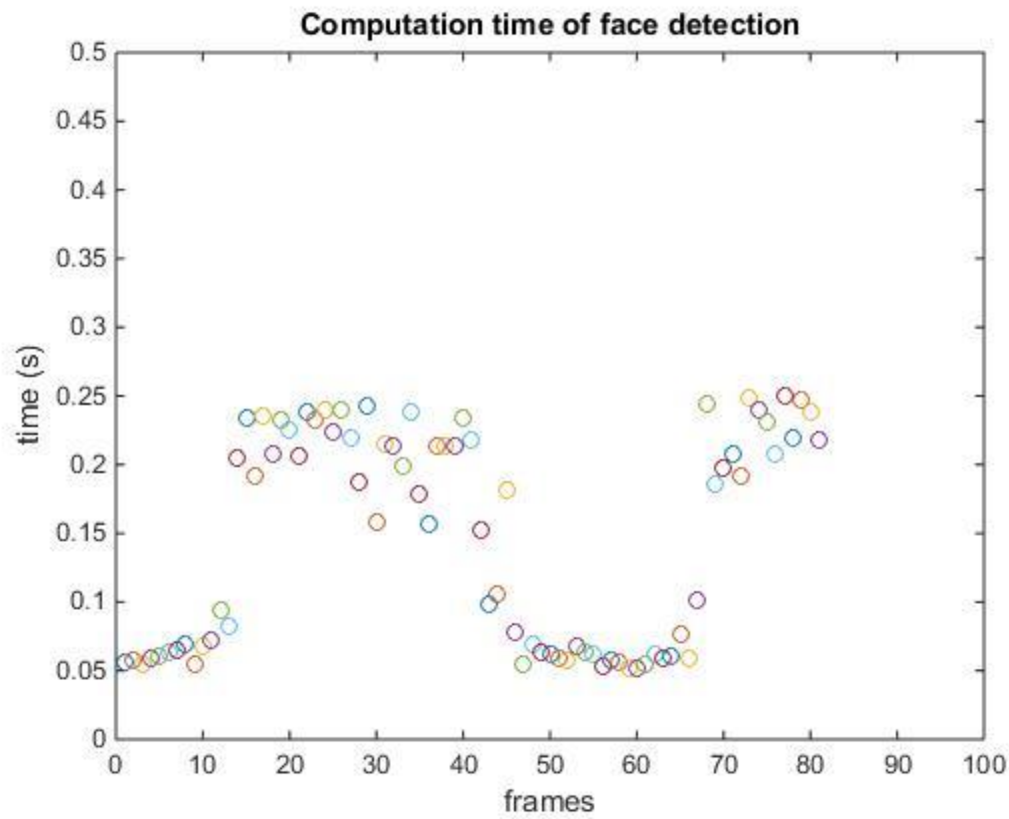
Nonetheless, the weak point of the application is its real-time capability. It cannot process frames at a quick pace. The reason is that the face detection part takes too much time.

Average for 81 frames

Face detected	0.2227 sec

Face non-detected	0.0834 sec

In the next graph, you can observe the time taken for the face detection only.



As you can see, there are some gaps in the graph. These are caused because the face detection time varies whether a face is detected or not. Actually, when a face is found, the time is longer. That is logical because in the case of "face found", the sub-window has been going through all the stages of the cascade classifier which takes some time whereas in the case of "no face found", the sub-window has been rejected from the beginning.

To resolve this issue, I modified some of the parameters:

- Decreasing the Image size
- Adapting the scale of the Haar-features to my application. Since I want to recognize only the face of the person in front of the camera, I do not need to take care of small scales which are useful to detect small faces of people far from the camera in the background for example.

- Limit of found faces. The application should find only one face and stops when one face is found.

By tuning them, I could improve the performance drastically but not to a stage where it can process in real-time. According to my colleagues, the problem comes from Matlab software. Indeed, Matlab may be not suited for real-time application and using C++ along with OpenCV library would be more appropriate.

V. Reflection

In this chapter, I will reflect the situations, tasks, results that I have experienced during my internship.

Working here as a research student helped me develop and clarify lots of things. Being completely inexperienced in the field of computer vision, I had to start from the beginning by learning elementary techniques and concepts of that field. Of course, the area of computer vision being very broad, I could not cover them all. Now, I am able to read some research papers involving concepts of computer vision.

Reading research papers was a very hard task for me, the contents are very dense and it is easy to encounter concepts that you don't know, making it harder to understand. Even though at the beginning, it was difficult for me to understand lots of papers, I managed to make up for not knowing certain mathematical and computer vision concepts that were explained in those papers. Indeed, I realized that mathematics is the base of it, linear algebra and statistics in particular.

Using Matlab during my internship helped me be more familiar with it and I am now used to work with that software.

Having done this internship helped me clarify some thoughts that I had before coming to Japan.

First, after learning computer vision for the first time and being able to attend the laboratory seminars, I understood that I wanted to continue study that subject. I was already interested before starting the internship but I had only an overview of it since I did not dive in that area before. But through my colleagues' work, I saw many applications of computer vision like for the classification of protein structures, or motion recognition which are not very known by the public but which will be without doubt useful in the future. That's why I have the intention of pursuing my studies at UTBM by choosing computer vision classes next semester.

Then, even though I am highly interested in working in the field of new technologies, I realized that research does not fit me. The truth is that working alone and doing pure research is not something I would enjoy. Indeed, I really found my colleagues' presentations very interesting and very clear but personally, I would prefer working in a more concrete project and with a team. During my internship, I did not do research since I developed an application for face recognition by using the Subspace methods but I could observe my colleagues' way of working.

Finally, working abroad is now something I would consider greatly. I had the opportunity to talk and to work with a lot of foreign people as a part of internship and outside it. That was really enjoyable and I did not have lots of difficulties communicating with them and expressing my ideas (when we could communicate in English). My Japanese language skills were high enough to be able to communicate in daily conversations but when it was about technical things, it was more complicated. However, I still have the firm intention of continuing to study that language because my interest of Japan goes beyond the area of work.

VI. Conclusion

My stay in Japan has been very instructive for me. Professor Fukui has offered me the opportunity to dive into the area of computer science with a very interesting project, the development of a face recognition system. I did not know at all how such a system worked, how could it recognize faces from only computer data.

Now, a lot of things has become clearer in my mind and I could actually put in application what I learn from the university to build the project.

I finally managed to build a real-time face recognition system based on Subspace methods with fair results, the only weakness being the computation time of the face detection algorithm. The best way would be to use C++ along with OpenCV library but because of the lack of time, it was not possible to implement it in C++ as well.

Besides, my Japanese language skills as well as my English language skills have improved considerably. Writing reports every month has been an effective practice for my writing skills. The international experience I gained here is invaluable and will be helpful for my future career and my personal life.

This internship was definitely beneficial for me and I'm grateful and thankful that I got to experience and learn many things.

VII. Bibliography

Subspace Methods

[1] Kazuhiro Fukui, Osamu Yamaguchi,
'Comparison between Constrained Mutual Subspace Method
and Orthogonal Mutual Subspace Method' (2006),

[2] Kazuhiro Fukui, Osamu Yamaguchi,
'Face Recognition Using Multi-view point Patterns for
Robot Vision' (2003),

[3] Kazuhiro Fukui, Osamu Yamaguchi, Bjorn Stenger,
'A framework for 3D object recognition using the kernel constrained mutual
subspace method' (2006)

[4] Kazuhiro FUKUI, Osamu YAMAGUCHI,
'The Kernel Orthogonal Mutual Subspace Method and its Application to 3D Object
Recognition' (2007)

[5] Kazuhiro Fukui,
'Subspace Methods'

[6] Hitoshi Sakano,
'A Brief History of the Subspace Methods'

[7] Gregory Shakhnarovich, Baback Moghaddam,
'Face Recognition in Subspaces' (2004)

[8] Masashi Nishiyama, Osamu Yamaguchi, Kazuhiro Fukui,
'Face Recognition with the Multiple Constrained Mutual Subspace Method' (2005),

[9] Yasuhiro OHKAWA, Kazuhiro FUKUI,
'Hand Shape Recognition based on Kernel Orthogonal Mutual Subspace Method'
(2009)

[10] E. Oja, 'Subspace methods of Pattern Recognition'

Face Detection

[11] Paul Viola, Michael Jones,
'Rapid Object Detection using a Boosted Cascade of Simple Features' (2001)

[12] Ruslana Makovetsky, Gayane Petrosyan,
'Face Detection and Tracking with Web Camera' (2012)

Kernel Methods

[13] Ming-Hsuan Yang,
'Kernel Eigenfaces vs Kernel Fisherfaces: Face recognition using kernel methods' (2002)

[14] Ming-Hsuan Yang,
'Face recognition using kernel methods' (2001)

[15] Daniel Olsson,
'Applications and Implementation of Kernel Principal Component Analysis to Specific Data Sets Master's Thesis Report' (2011)

[16] Bernhard Scholkopf, Alexander Smola, Klaus-Robert Muller, 'Nonlinear Component Analysis as a Kernel Eigenvalue Problem Technical Report' (1996)

Others

[17] Hiroshi Murase, Shree K.Nayar,
'Parametric Eigenspace Representation for Visual Learning and Recognition' (1993)

[18] Marian Stewart Bartlett, Javier R. Movellan, Terrence J. Sejnowski,
'Face Recognition by Independent Component Analysis' (2002)

[19] Xiao Zhang, Lei Zhang, Xin-Jing Wang, Heung-Yeung Shum, 'Finding Celebrities in Billions of Webpages' (2012), MSRA-CFW Database