ECE533 – Image Processing Project

# Face Recognition Techniques

THE UNIVERSITY of WISCONSIN MADISON

Jorge Orts

# Index

# INTRODUCTION

This project deals with the topic of face recognition techniques using digital image processing. Face recognition has always been a very challenging task for the researches. On the one hand, its applications may be very useful for personal verification and recognition. On the other hand, it has always been very difficult to implement due to all different situation that a human face can be found.[6] Nevertheless, the approaches of the last decades have been determining for face recognition development. Due to the difficulty of the face recognition task, the number of techniques is large and diverse. In addition, the applications involve a huge number of situations.

Although we can find many other identification and verification techniques, the main motivation for face recognition is because it is considered a passive, no intrusive system to verify and identify people.[3] There are many other types of identification such as password, PIN (personal identification number) or token systems. Moreover, it is nowadays very instilled the usage of fingerprints and iris as a physiological identification system. They are very useful when we need an active identification system; the fact that a person has to expose their body to some device makes people feel being scanned and identified. The pause-and-declare interaction is the best method for bank transactions and security areas; people feel conscious of it, as well as comfortable and safe with it. However, we do not want to interact with people that way in many other applications that required identification. For example, a store that wishes to recognize some customers or a house that has to identify people that live in there. For those application, face as well as voice verification are very desirable. It is also important that an identification technique is closer to the way human beings recognize each other. [5]

As it has already said previously, the applications for face recognition are very varied. We can divide them into two big groups, the applications that required face identification and the ones that need face verification. The difference is that the first one uses a face to match with other one on a database; on the other hand, the verification technique tries to verify a human face from a given sample of that face.[6] Face recognition could be also divided into two different groups, according to their field of application. The main reason for promoting this technique is law enforcement application; however, it can also be used for commercial application. Among the law enforcement applications, some representative examples are mug shot albums, video surveillance and shoplifting.[3] Concerning commercial applications we can differentiate between entertainment (video games, virtual reality and training programs), smart cards (driver's license, passport and voter registration) and information security (TV parental control, cell phone and database security).[7]

It has already been stated that face recognition techniques have always been a very challenging task for researches because of all difficulties and limitations. Human faces are not an invariant characteristic; in fact, a person's face can change very much during short periods of time (from one day to another) and because of long periods of time (a difference of months or years). One problem of face recognition is the fact that different faces could seem very similar; therefore, a discrimination task is needed. On the other hand, when we analyze the same face, many characteristics may have changed. Ones of the most important problems are changes in illumination, variability in facial expressions, the presence of accessories (glasses, beards, etc); finally, the rotation of a face may change many facial characteristics. [6]

## APPROACH

The paper principally deals with the comparison of two different methods for face recognition. The project is based on two articles that describe these two different techniques; they are attached at the references as source [3] and [4]. These methods are *"Face Recognition Using Eigenfaces"* and *"Face recognition using line edge map"*.

For each of the techniques, a short description of how it accomplishes the described task will be given. Furthermore, some tables and results will be showed in order to understand the accuracy of each method. This report only shows a comparison of already made research studies; therefore, the pictures used and the data are extracted from the original sources. Since the methods do not follow a common line, they will be described separately; it is due to the different ways to achieve face recognition. We could divide face recognition techniques into two big groups: the ones that tackle the problem using geometric approach, and the ones that use feature-based characteristics. [6]

Finally, a discussion about the best characteristics of each method will be carried out. Depending of the technique, and more important of the work performed to make the article, different situation of face position, lighting, etc will be commented. The main goal of this paper is find a good face recognition technique depending on the situation.

# WORK PERFORMED

During this section, the two face recognition algorithms will be explained in order to understand the basis of each one. They have been selected because two main reasons: they are very known and spread techniques for face recognition; moreover, they represent different ways to approach the problem as it was stated before. The first technique is based on the so-called Karhunen-Loève transformation using eigenfaces for recognition. The second one tries a new algorithm using line edge maps to improve the previous methods such as the eigenfaces.

## Face recognition using eigenfaces

As a general view, this algorithm extracts the relevant information of an image and encodes it as efficiently as possible. For this purpose, a collection of images from the same person is evaluated in order to obtain the variation. Mathematically, the algorithm calculates the eigenvectors of the covariance matrix of the set of face images. [4]

Each image from the set contribute to an eigenvector, these vectors characterize the variations between the images. When we represent these eigenvectors, we call it eigenfaces. Every face can be represented as a linear combination of the eigenfaces; however, we can reduce the number of eigenfaces to the ones with greater values, so we can make it more efficient. The basic idea of the algorithm is develop a system that can compare not images themselves, but these feature weights explained before. The algorithm can be reduced to the next simple steps.

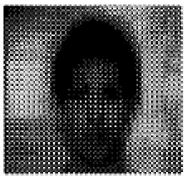
1. Acquire a database of face images, calculate the eigenfaces and determine the face space with all them. It will be necessary for further recognitions.

2. When a new image is found, calculate its set of weights.

3. Determine if the image is a face; to do so, we have to see of it is close enough to the face space.

4. Finally, it will be determined if the image corresponds to a known face of the database of not.

(a)



(b)

**Figure 1: (a) set of face images. (b) Average of the set of images given above**

Let the training set of face images be $I_1, I_2, I_3 \ldots I_M$. We calculate the average of the set as $\Psi = \frac{1}{M}\sum_{N=1}^{M} I_n$. In addition, the difference of each image from the average is defined by $\Phi_i = I_i - \Psi$. We can see if Figure 1(a) a set of images and its average in Figure 1(b). Finally, we calculate the eigenvalues $\lambda_k$ and eigenvectors $u_k$ of the covariance matrix $C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T$. [4]

The last step is to classify a face image. We just need to transform the new image into its eigenfaces components; i.e. its project into face space. We have to calculate the vector of weights $\Omega^T = [\varpi_1, \varpi_2 .. \varpi_{M'}]$, where $\varpi_k = u_k^T (I - \Psi)$ for $k = 1,2...M'$; and $M'$ represents not the total eigenfaces, but the ones with greater values. The criterion to determine which the matched face image is is to determine the image face class k that gives the minimum Euclidean distance $\varepsilon_k = \|(\Omega - \Omega_k)\|$, where $\Omega_k$ is the vector that describes the face image number k. We can see an example of this procedure in Figure 2. Image (a) and (b) represents the case when the input image is near face space (it is a face) and a face class (face matched). On the other hand, image (c) shows an example of an input image distant from face space (in fact, it is a flower, not a human face) and not near a known face class. We could also find an input image that is not near face space but it still is near a face class; it would be detected as a false positive and it depends on the value of threshold set to compare the Euclidean distance explained previously.

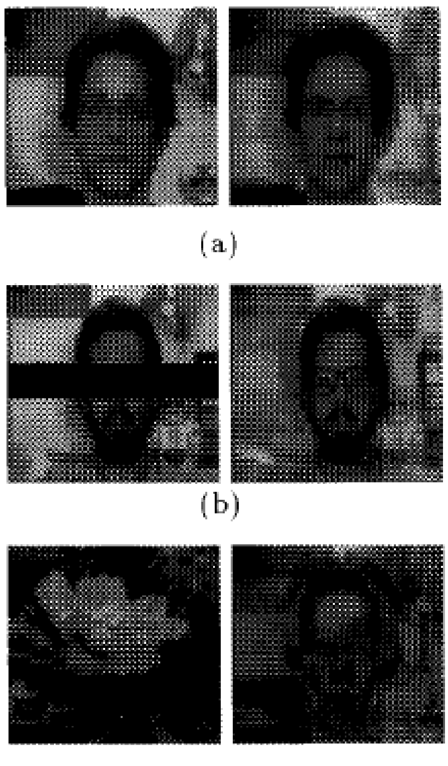


**Figure 2: Three examples of input images projected on the face space**

## Face recognition using line edge map

This algorithm describes a new technique based on line edge maps (LEM) to accomplish face recognition. In addition, it proposes a line matching technique to make this task possible. In opposition with other algorithms, LEM uses physiologic features from human faces to solve the problem; it mainly uses mouth, nose and eyes as the most characteristic ones.

In order to measure the similarity of human faces the face images are firstly converted into gray-level pictures. The images are encoded into binary edge maps using Sobel edge detection algorithm. This system is very similar to the way human beings perceive other people faces as it was stated in many psychological studies. The main advantage of line edge maps is the low sensitiveness to illumination changes, because it is an intermediate-level image representation derived from low-level edge map representation.[3] The algorithm has another important improvement, it is the low memory requirements because the kind of data used. In Figure 3, there is an example of a face line edge map; it can be noticed that it keeps face features but in a very simplified level.

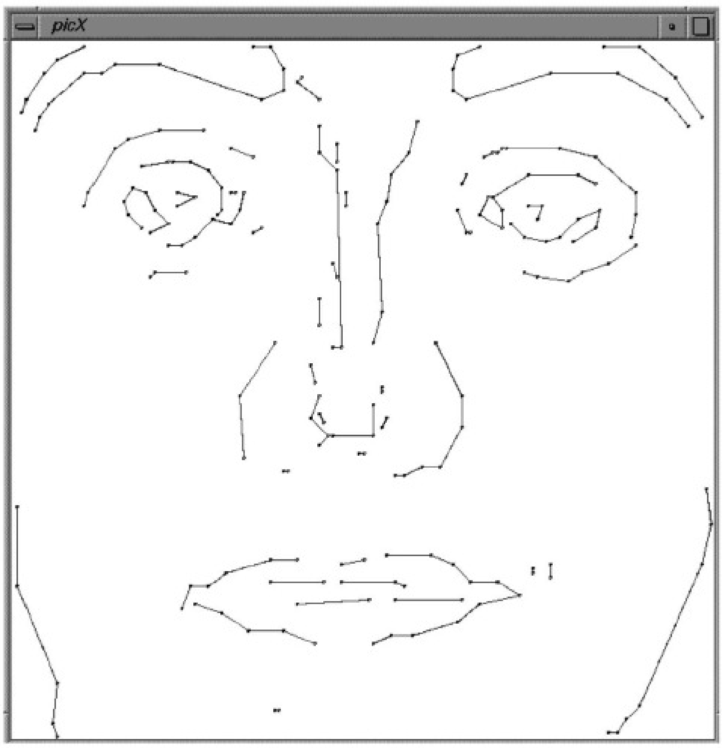



**Figure 3: Example of face LEM**

One of the most important parts of the algorithm is the Line Segment Hausdorff Distance (LHD) described to accomplish an accurate matching of face images. This method is not oriented to calculate exact lines form different images; its main characteristic is its flexibility of size, position and orientation. Given two LEMs $M^l = \{m_1^l, m_2^l ... m_p^l,\}$ (face from the database) and $T^l = \{t_1^l, t_2^l ... t_q^l\}$ (input image to be detected); the LHD is represented by the vector $\bar{d}(m_i^l, t_j^l)$. The elements of this vector represent themselves three difference distance measurements: orientation distance, parallel distance and perpendicular distance respectively.

$$\bar{d}(m_i^l, t_j^l) = \begin{bmatrix} d_\theta(m_i^l, t_j^l) \\ d_{\|}(m_i^l, t_j^l) \\ d_\perp(m_i^l, t_j^l) \end{bmatrix}$$

$$d_\theta(m_i^l, t_j^l) = f(\theta(m_i^l, t_j^l)) \qquad d_{\|}(m_i^l, t_j^l) = \min(l_{\|1}, l_{\|2}) \qquad d_\perp(m_i^l, t_j^l) = l_\perp$$

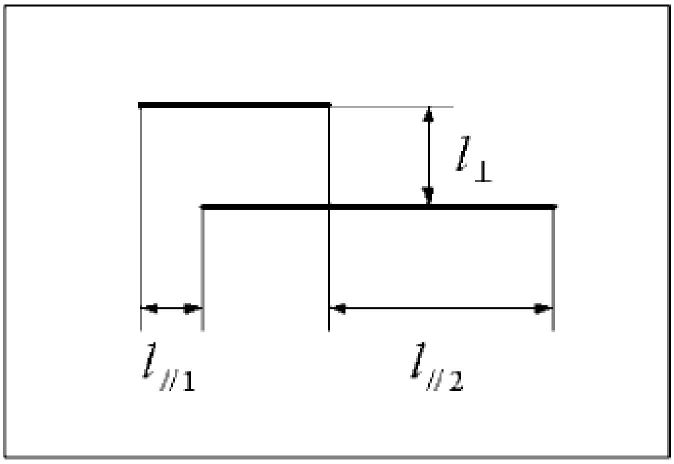



**Figure 4: Practical example for calculating parallel and perpendicular distances**

The function $\theta(m_i^l, t_j^l)$ represents the smallest intersection angle between lines $m_i^l$ and $t_j^l$. The function f is a penalty nonlinear function that ignores smaller angles and penalizes greater ones. It can be used $f(x) = x^2/W$ as the penalty function. How to calculate the parallel and perpendicular distance is shown in Figure 4. Finally, the distance between the two segments can be calculated with the next equation.

$$d\left(m_i^l, t_j^l\right) = \sqrt{d_\theta^2\left(m_i^l, t_j^l\right) + d_{\|}^2\left(m_i^l, t_j^l\right) + d_\perp^2\left(m_i^l, t_j^l\right)}$$

After having defined the distance between two lines, line segment Hausdorff distance (pLHD) is defined as

$$H_{pLHD}\left(M^l, T^l\right) = \max\left(h\left(M^l, T^l\right), h\left(T^l, M^l\right)\right)$$

where $h\left(M^l, T^l\right) = \frac{1}{\sum_{m_i^l \in M^l} l_{m_i^l}} \sum_{m_i^l \in M^l} l_{m_i^l} \min_{t_{ji}^l \in T^l} d\left(m_i^l, t_j^l\right)$ and $l_{m_i^l}$ is the length of segment $m_i^l$.

The main strength of this distance measurement is that measuring the parallel distance, we choose the minimum distance between edges. It helps when line edge is strongly detected and the other one not. It avoids shifting feature points. However, it also has a weakness; briefly, it can confuse lines and not detect similarities that should be detected. In order to avoid errors, another measurement can be made. We can add a new parameter to the Hausdorff distance, comparing the number of lines in the images is a good method to exclude images. [3]

# RESULTS

Since the objective of this project is not the implementation of the algorithms, but the description and comparison of them, the results will be reported from the experiments performed by the authors of the articles. Both methods were tested using variations of face orientation, illumination and size.

## Face recognition using eigenfaces results

The eigenfaces algorithm used a database of 2500 face images taken from 16 subjects. Each subject was exposed to all combinations of three head orientations; moreover, a six level Gaussian pyramid was created so each image had resolutions from 512x512 to 16x16 pixels. Two different experiments were performed, the first one allowed an infinite value of the threshold $\theta_\in$. On the other hand, the second experiment varied this threshold in order to achieve conclusions about it.

During the first experiment no face was rejected as unknown because of the infinite threshold, statistics were collected measuring the mean accuracy as a function of the difference between the training conditions and the test conditions.[4] The results were a 96% of accuracy with illumination changes, 85% with orientation variation and a 64% when the sized changed.

The second experiment tried both a low threshold and a high one in order to compare the accuracy of recognition and the rejected images. With a low value of $\theta_{\in}$, many images were rejected because they were considered not belonging to the database; however, a high correct classification percentage was achieved. On the other hand, using a high value of $\theta_{\in}$, the great majority of images were accepted but the errors increased. Finally, adjusting the threshold to obtain a 100% of recognitions, the unknown rates were 19% for lighting variations, 39% for orientation and 60% for size. If $\theta_{\in}$ was set to obtain only a 20% of unknown rate, correct recognitions were 100%, 94% and 74% respectively.[4]

**<u>Face recognition using line edge map results</u>**

The images for the experiments belong to three different databases, University of Bern for pose variations, the AR database from Purdue University was used to evaluate the algorithm with illumination and size variations, The Yale face database had the purpose of compare the algorithm with other methods. The experiments were performed using three different algorithms: the edge map, the eigenfaces and LEM. Therefore, tables with a comparison of the algorithms are provided.

Tables with the corresponding results are shown in order to make a good comparison with the other algorithm discussed in this paper. The results show probabilities of correct detection of the different algorithms and some experiments include variation of parameters such as number eigenvectors or light direction.

Each table is labeled so its content can be understood. Not all the results from the article are showed in this project, but it has the necessary to make a good comparison of the general characteristics of the algorithm.

| | Bern database | | | AR database | | |
|---|---|---|---|---|---|---|
| Method | EM | Eigenface | LEM | EM | Eigenface | LEM |
| Recognition rate | 96.7% | 100% | 100% | 88.4% | 55.4% | 96.4% |

**Table 1: Results for size variations for edge map, eigenface (20 eigenvectors) and LEM.**

| Method | Recognition rate |
|---|---|
| LEM | 96.43% |
| Eigenface (20-eigenvectors) | 55.36% |
| Eigenface (60-eigenvectors) | 71.43% |
| Eigenface (112-eigenvectors) | 78.57% |

**Table 2: Comparison of LEM and eigenfaces methods with the AR database images**

| Testing faces | Eigenface | | Edge map | LEM |
|---|---|---|---|---|
| Left light on | 20-eigenvectors | 6.25% | 82.14% | 92.86% |
| | 60-eigenvectors | 9.82% | | |
| | 112-eigenvectors | 9.82% | | |
| | 112-eigenvectors w/o 1$^{st}$ 3 | 26.79% | | |
| Right light on | 20-eigenvectors | 4.46% | 73.21% | 91.07% |
| | 60-eigenvectors | 7.14% | | |
| | 112-eigenvectors | 7.14% | | |
| | 112-eigenvectors w/o 1$^{st}$ 3 | 49.11% | | |
| Both lights on | 20-eigenvectors | 1.79% | 54.46% | 74.11% |
| | 60-eigenvectors | 2.68% | | |
| | 112-eigenvectors | 2.68% | | |
| | 112-eigenvectors w/o 1$^{st}$ 3 | 64.29% | | |

**Table 3: Results for lightning variation for three algorithms**

| | Recognition rate | | | |
|---|---|---|---|---|
| Method | Edge map | Eigenface (20-eigenvectors) | Eigenface (30-eigenvectors) | LEM |
| Looks left/right | 50.00% | 70.00% | 75.00% | 74.17% |
| Looks up | 65.00% | 51.67% | 56.67% | 70.00% |
| Looks down | 67.67% | 45.00% | 55.00% | 70.00% |
| **Average** | **58.17%** | **59.17%** | **65.12%** | **72.09%** |

**Table 4: Results for the three algorithms and different face poses**

## DISCUSSION

Since results for the eigenfaces algorithm are reported by both articles, this project will take conclusion from the results showed in [4]. The main reason is that the results from [3] of the eigenfaces method do not describe the exact procedure; therefore, the reported results will not be as reliable as expected. Other reason is that an article that tries to demonstrate that LEM algorithm is better than others like eigenfaces might not be as objective as [3].

The first conclusion that can be said from the results of the eigenfaces algorithm is related to the threshold to determine a match in the input image. It was demonstrated that the accuracy of recognition could achieve perfect recognition; however, the quantity of image rejected as unknown increases. The dependence of accuracy and features changing is other characteristic to take into account. The results show that there is not very much changes with lighting variations; whereas size changes make accuracy fall very quickly. In order to avoid the most important weakness a multiscale approach should be added to the algorithm. [4]

As it was predicted, for lighting variations the LEM algorithm kept high levels of correct recognitions. In addition, LEM method always managed the highest accuracy compared with eigenfaces and edge map. The disagreement between two articles about the results of eigenfaces with lighting variations could be due to a matter of concept. [4] took into account only the images recognized as faces to test the algorithm; however, [3] might have considered also the rejected images to make the statistics.

LEM algorithm demonstrated a better accuracy than the eigenfaces methods with size variations. While eigenfaces difficultly achieved an acceptable accuracy, LEM manage to obtain percentages around 90%, something very good for a face recognition algorithm. Finally, taking into account the results from [4] for orientation changes, LEM algorithm could not beat eigenfaces method. LEM hardly reach a 70% for all different poses.

As a general conclusion, it could be said that LEM, as a more recent research; allows better results for lighting and size variations. More concretely, it beats eigenfaces method with size variation; where it has its most important weakness. On the other hand, eigenfaces algorithm demonstrated better results for posing changes than LEM, possibly because of the basis of the algorithm. LEM is based on face features, while eigenfaces uses correlation and eigenvector to do so.

# REFERENCES

[1] L.C. Jain, *"Intelligent biometric techniques in fingerprint and face recognition"*. Boca Raton: CRC Press, 1999.

[2] Aurélio Campilho, Mohamed Kamel, *"Image analysis and recognition : international conference, ICIAR 2004, Porto, Portugal, September 29-October 1, 2004"*. Berlin ; New York : Springer, c2004.

[3] Yongsheng Gao; Leung, M.K.H., *"Face recognition using line edge map"*. Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 24 Issue: 6 , June 2002, Page(s): 764 -779.

[4] M.A. Turk, A.P. Pentland, *"Face Recognition Using Eigenfaces"*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3-6 June 1991, Maui, Hawaii, USA, pp. 586-591.

[5] Pentland, A.; Choudhury, T. , *"Face recognition for smart environments "*. Computer, Volume: 33 Issue: 2 , Feb. 2000, Page(s): 50 -55.

[6] De Vel, O.; Aeberhard, S., *"Line-based face recognition under varying pose"*. Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 21 Issue: 10 , Oct. 1999, Page(s): 1081 -1088.

[7] W. Zhao, R. Chellappa, A. Rosenfeld, and J. Phillips, *"Face Recognition: A Literature Survey"*. ACM Computing Surveys, Vol. 35, No. 4, December 2003, pp. 399–458.

[8] Face recognition home page: http://www.face-rec.org/