Face Recognition: Robustness of the 'Eigenface' Approach

Carmen Au[1], Jean-Sebastien Legare[2] & Reehan Shaikh[2]

McGill University, Montréal, Québec, Canada
[1]School of Computer Science & Center for Intelligent Machines
{au}@cim.mcgill.ca
[2]School of Computer Science
{jlegar, reehan.shaikh}@cs.mcgill.ca

Abstract

While face recognition is a fairly trivial task for humans, much of computer vision research has been dedicated to finding an algorithm to teach a computer how to recognize faces. This paper discusses the robustness of the Turk and Pentland 'Eigenface' algorithm [1]. The algorithm consists of two stages, the learning stage, which is done offline, and the recognition stage, which is done online. The learning stage consists of making a database of the principal components of all the images in the training set to which new images can be compared to. This database is called the "face space". The recognition stage projects each new image of a face onto the "face space", using principal component analysis, and compares it to known faces from the training set to find the best match. The algorithm has a high recognition rate when the images of the faces were upright and had similar lighting and feature conditions to the training images. By feature conditions, we mean smiling, winking, glasses, etc… However, once the faces were tilted, the lighting was changed or the face was obscured somehow, the algorithm suffered from a loss of recognition.

1 Introduction

For humans, recognizing faces is a relatively simply task. Faces can be partially occluded or rotated in various directions without too much loss of recognition. A major area of computer vision research is in the automation of video surveillance. If this automation is to be achieved, then finding a fast and computationally efficient face recognition algorithm is essential. Thus, there has been a plethora of papers written on this subject. Many of the proposed algorithms use a feature-based approach [2] to recognition. The feature-based algorithms look at major features of the face and compare them to the same features on other faces. Some of these features include the eyes, ears, nose and mouth. The approach uses the position, size and relationship of these facial features to perform the comparisons. Other algorithms use the 'Connectionist' approach [3]. This approach uses a general two-dimensional pattern of the face and neural networks for recognition. The 'Connectionist' approach often requires a large number of training faces to achieve decent accuracy. Finally, Kirby and Sirovich presented the 'Eigenface' approach [4], after which, many papers have been written on their basic idea. This paper will discuss the implementation of one such algorithm and attempt a critique on whether or not it is a viable solution for real-time applications.

In [1], Turk and Pentland suggest an algorithm that treats the face recognition problem as a two-dimensional problem, which assumes that most faces are under similar conditions. The underlying idea in their algorithm is that images of faces are compared to those of known faces and whether the face matches to one of the known faces or if it is a new face or if it is not a face at all is determined. Before these comparisons can be done, a database of known faces is required. Rather than storing all the image information of the entire set of known faces, the Turk and Pentland algorithm suggests a method which stores only the "eigenfaces". These are the eigenvectors of the set of faces. All the "eigenfaces" combined make up a "face space" and each new face is projected onto this "face space". A detailed explanation of the algorithm is found in section 2 of this paper.

While under strict, uniform and artificial conditions, many of the aforementioned algorithms could produce effective results. Nonetheless, under real-time conditions, these algorithms may not produce the results that we are after. This paper will discuss the robustness of the Turk and Pentland algorithm and how it fares as a real-time video surveillance system. Intuitively, the assumption that all faces exhibit similar conditions is a fair assumption. However, even under similar conditions, there are many angles at which the head could be tilted or directions in which the faces could be turned or features that can be varied. While humans would still be able to recognize a face which is rotated, say 50º, to the right or left, this is not a trivial task to teach a computer. Thus, this paper will discuss tests which attempt to see how much the face could be rotated or tilted before the algorithm breaks down. Moreover, often it is difficult to obtain clear unobstructed views of the faces. The algorithm will be tested with images of known faces from the database but that are obscured at varying degrees to see how much, if any, obscurity is allowable. Changes in illumination, translation of images and resolution changes are also conditions that will be tested.

2 Algorithm

Under the assumption that human faces are similar, it turns out that any face image can be encoded as a combination of feature images, each of which captures one "direction" of the variability of faces. The idea behind the algorithm proposed by Turk and Pentland is to extract only the relevant information of a face image, encode it as efficiently as possible and compare that encoding to a database of models encoded similarly. In mathematical terms, the algorithm finds the principle components of the distribution of faces. These principle components can be thought of as the set of features which together characterize the variation between the faces. They may not be necessarily related directly to our intuitive notion of features such as the eyes, lips, nose, and hair; but rather related to the variation of intensity in respective sample points of different face images. Each image location contributes more or less to each principal component, so that the latter can be displayed as a sort of ghostly face, called an "eigenface".

It is possible to represent exactly each image in the training set in terms of a linear combination of the "eigenfaces". The number of possible "eigenfaces" is equal to the number of face images in the training set. However, using principle component analysis, it is possible to approximate the faces using only the *best* "eigenfaces", that is, the ones that account for the most variation. The following steps summarize the recognition process:

1. Initialization: Acquire the training set of face images and calculate the "eigenfaces", which define the "face space".

2. Projection: When a new image is encountered, calculate the set of weights based on the input image and the M "eigenfaces" by projecting the input image onto each of the "eigenfaces".

3. Detection: Determine if the image is a face at all (whether known or unknown) by checking if the image is sufficiently close to the "face space".

4. Recognition: If it is a face, classify the weight pattern as either a known or unknown person.

5. Learning (optional): If the same unknown face is seen several times, calculate its characteristic weight pattern and incorporate it into the known faces.

2.1 Calculating the 'Eigenfaces'

Let each face image, $I(x, y)$, be a two-dimensional $w \times h = P$ array of intensity values. Each image, written as a $P \times 1$ vector, represents a point in P-dimensional space. Therefore, the collection of images in the training set constitutes a collection of points in a huge space. Again, due to the similarity of faces in their overall configuration, these points will not be randomly distributed in this immense space, but are likely to be close and occupy only a small portion of the space. Thus, the collection of points can be described by a relatively low dimensional subspace.

Let the training set of images be $\Gamma_1, \Gamma_2, ..., \Gamma_M$. The average face of this set is then defined by

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

Now, let each face differ from the mean face by $\Phi_i = \Gamma_i - \overline{\Gamma}$. For each location in an image, we have one sample for each of the *M* images. We can study the intensity relations between any two sample points by analyzing their covariance. The covariance between points *i* and *j*, denoted c_{ij}, can be approximated by

$$c_{ij} = \frac{1}{M}\sum_{k=1}^{M}\Phi_k(i)\Phi_k(j)$$

Therefore, the covariance matrix *C* can be obtained as follows

$$C = AA^T = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T$$

where $A = [\Phi_1 \quad \Phi_2 \quad \dots \quad \Phi_M]$. By using principle component analysis analysis on the set of large vectors, we have obtained a set of *M* orthonormal vectors $\vec{u}_n$ and their associated eigenvalues λ_n, which best describe the spread of the data in the *P*-dimensional space. These orthonormal vectors, i.e. the principle components, turn out to be the eigenvectors of the covariance matrix *C*.

The matrix *C*, however, is $P \times P$ and determining the *P* eigenvectors and eigenvalues is an intractable task for typical image sizes (usually greater than or equal to 256 × 256). Nevertheless, when *M* is very small compared to *P*, like the case in our experiments, a smaller $M \times M$ problem can be solved instead. Consider the eigenvectors v_i of $A^{\mathrm{T}}A$ such that

$$A^{\mathrm{T}}Av_i = \mu_i v_i$$

Multiplying each side from the right by A yields

$$AA^{\mathrm{T}}Av_i = \mu_i Av_i$$

From this we see that Av_i are the eigenvectors of C. If we let $V = [v_1 \quad v_2 \quad ... \quad v_M]$ be the matrix formed from the eigenvectors of $A^{\mathrm{T}}A$ and $U = [u_1 \quad u_2 \quad ... \quad u_P]$ be the matrix formed from the eigenvectors of AA^{T}, then $U = AV$.

Although *M* eigenvectors ("eigenfaces") are necessary to encode each image of the training set without loss of information, *M' < M* are sufficient enough for recognition. Therefore, from the *M* eigenvectors of *V*, we pick the *M'* eigenvectors that account for the most variation, i.e. the *M'* eigenvectors having the highest eigenvalues.

The number of significant "eigenfaces" to consider can be picked arbitrarily. Several criterions have been established in the past as solutions to the "number-of-factors" problem. The Kaiser criterion [5] for instance, which selects only eigenvectors whose values are above 1, seems to be the most widely used. Instead, for our tests, we have determined *M'* with a threshold Θ_λ. This is a ratio of the summations of the eigenvalues, computed as follows

$$M' = \min_r\left\{ r \mid \frac{\sum_{i=1}^{r}\lambda_i}{\sum_{i=1}^{M}\lambda_i} > \Theta_\lambda \right\}$$

2.2 Identifying Faces

Once the basis vectors for the "face space" have been constructed, all that remains is to project all the images in the training set onto the "face space". Any image Γ can be expressed in terms of the *M'* "eigenfaces", using *M'* weights calculated as follows

$$\omega_k = u_k^{\mathrm{T}}\left(\Gamma - \overline{\Gamma}\right)$$

These *M'* weights form a vector $\Omega^T = [\omega_1 \quad \omega_2 \quad \ldots \quad \omega_{M'}]$, quantifying the contribution of each of the "eigenfaces" in representing the input face image, treating the "eigenfaces" as a basis set for the face images. The weight vector is then used to determine which of the predetermined number of faces matches best the query image. The easiest method to identify an image in the training set that provides the best description of the input image is to choose the face image that minimizes the Euclidean distance between weight vectors, that is

$$\varepsilon_k^2 = \left\|\left(\Omega - \Omega_k\right)\right\|^2$$

where Ω_k is the vector describing the kth image in the training set. The algorithm proposed by Turk and Pentland makes the distinction between a "face class" and a face image. A "face class" consists of the collection of face images belonging to an individual, and in the case where a "face class" contains more than one image, Ω_k is calculated as the average of the weights obtained when projecting each image of a class *k*. For reasons that we will explain soon, we chose to have only one image per "face class". The query image belongs to an individual *k* in the training set only when the minimum ε_k is below a threshold Θ_ε. On the other hand, if the minimum distance is above this fixed threshold, then the queried face is classified as unknown to the system.

The validity of the threshold relies on the assumption that faces from the same person map close to each other in the "face space". In other words, it relies on the assumption that the "face space" consists of a series of small clusters distant from each other, with each cluster representing faces from one individual, and where each cluster approximately has the same dimensions.

However, several times in our experiments, two sample faces from two different people mapped closer onto the "face space" than two faces from the same person, thus breaking the cluster assumption. Therefore, we could not establish the face identification threshold as proposed originally. Images from the same individual seemed scattered across the "face space", hence we chose to only put one image per face class. We fixed the value of the threshold "instinctively" so as to get a reasonable ratio of correct/incorrect positive identifications.

Any image, given it has the right dimensions, can be projected onto the "face space". More specifically, any input image can be more or less approximated as a linear combination of the "eigenfaces". Since the *M'* largest eigenvectors were chosen to span the *M'*-dimensional "face space", they capture the most variation for the face images. This implies that projecting any non-face image onto this *M'* subspace is likely to yield weights for which no face would have mapped to.

Hence another distance measure is employed to determine whether the input image is a face or not. We can calculate the distance ε between the projection Ω of a query image and the average projection $\overline{\Omega}$ of all images in the training set. Mathematically,

$$\varepsilon^2 = \left\|\Omega - \overline{\Omega}\right\|^2$$

The projection weight vectors of the training set can be seen as a set of points in *M'*-space forming an *M'* "sphere", in which $\overline{\Omega}$ is the center. The radius of that "sphere" corresponds to the largest distance between a point and the center,

$$r = \max_i \left(\left\|\Omega_{,i} - \overline{\Omega}\right\|\right)$$

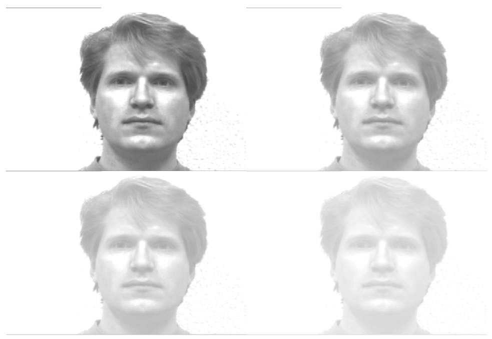
where i = 1, 2, …, *N*. We use this radius as a determining factor for the limits of the "face space". Any projection farther than *r* from the center is classified as not a face. Therefore, if the distance ε is smaller than the radius, we assume that the input image is a face. Otherwise, we assume that the image is something else.

3 Results and discussion

Our biological vision system seamlessly recognizes faces, whether they are partially occluded, tilted or rotated a certain amount of degrees or in various lighting conditions. Even if the background is noisy or the human visual system hasn't encountered a specific face in a very long time (i.e. the face has changed drastically over time), we can still distinguish and match the "query image", so to speak, to the vast database of images stored in our memory system, specifically in our long-term memory, involving the hippocampus and the temporal lobes of the brain. On the other hand, not only does a computer not have enough physical memory to compete with that of the brain, it doesn't even come close to the capacity and efficiency of face recognition as the brain. Here, we shall discuss the actual implementation of the 'Eigenface' approach. We show what happens if this approach was taken and implemented into a real-time system. Though the following query images aren't taken from a real-time system, the properties of these images are comparable to those of real-time system. Increases in brightness, occlusions of the face and noisy backgrounds are all part of real-time systems. We simulate these conditions in our images and run the images as queries. We present concrete examples to back up our findings. We use specific test cases, as discussed above, to break the proposed algorithm.

3.1 Brightness

Our first example focuses on brightness. In everyday life, we encounter faces in the bright sun or in the dark night and yet, we recognize without effort who it is that we see. Yet, slight changes in brightness give the computer a very hard time to match and recognize images. The main reason for this is that for a computer, an image is just an array of intensity values. Once these values are altered, the comparison is compromised greatly. Following are four images of varying brightness, of which the top left one is included in the training set, thus it are in the database of images. We try to match the other three to the top left one.



Of these, the top left image returns the proper answer, itself. The other three return as "not-a-face". The brightness is too high for the images to be mapped to the "face space".

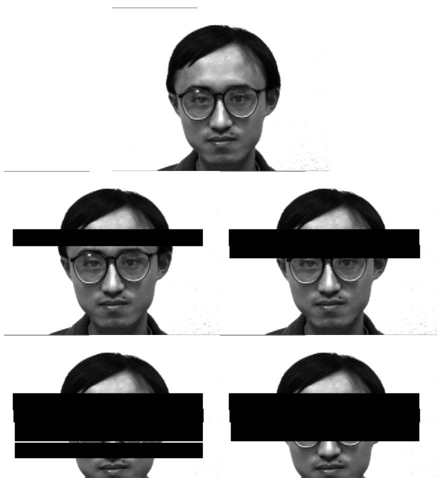
3.2 Varying conditions

The second example looks at what the outcome is if one training image in the database is taken under very different conditions. For example, the size of the face is too big with respect to the size of the image or lighting conditions aren't the same as every other training image. To the right are some images of such examples. If we train the database with these irregular images, as well as the regular images, the "face space" becomes very large, to the point where non-faces are mapped to the "face space". For example, a soft-drink can or an animal (with proper orientation so as to resemble a face) will be mapped and a corresponding face returned. This results in a very different "face space". To mathematically show that the distance between the average face and the

farthest face when including the irregular images is much larger than when the irregular images aren't included, we calculate the Euclidean distance. This Euclidean distance depicts the radius (and the variation) of the "face space". Since there is an irregular face with much more variability than the average face, the Euclidean distance becomes drastically large between the two. Thus, the learning process for this algorithm must be strongly controlled. Images taken for training must be taken under static conditions and outliers must be discarded or re-photographed. Otherwise, the algorithm runs very poorly. This is a very important concept for the Artificial Intelligence community who are in the works of face recognition robotics and automated machinery for this task.

3.3 Partial Occlusion

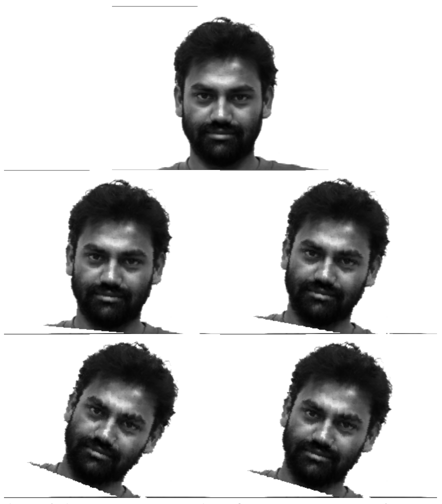
Imagine playing hide'n'seek with many people. You are looking for one of your friends and you spot him/her while he/she is looking away from you. In a split second, you know which friend it is, even though the majority, if not all, of your friend's face is occluded from your vision. Better yet, get a picture of your favourite get-together and look at the face that is occluded the most. Again, in a split second, you can recognize who it is that is occluded in the picture. These, among others, are great examples of how the human visual system handles occlusion so well. To a certain degree of occlusion, we recognize with ease. The following example shows how well the algorithm handles occlusion. Consider the following pictures. The first picture on the top is the original picture in the database of known faces. We added rectangles to the images following the original one to test how the algorithm handles occlusion. Surprisingly, the top left, top right, and bottom right pictures involving occlusion map to the original image.



This is a very impressive result. The fourth, bottom left image fails to map to the right person. This is directly related to the amount of occlusion introduced in the image. The more occlusion, the more information about the face is lost.

3.4 Tilting of the face

Now we shall test how orientation of the face affects the algorithm. We take an image and simply rotate it clockwise or counter clockwise. We see that the algorithm also performs remarkably well on this variation. As above with partial occlusion, there is a certain amount of variation that one can introduce before the algorithm breaks down. Consider the following images. Once again, the first picture on the top is the original image in the database of known faces. We rotate the image clockwise in increments of 5^0 in every successive image, clockwise from the top left to the bottom right. Again, the top left, top right and bottom right images which were rotated mapped to the original image. The last, bottom left image failed to map to the right person. This is also directly related to the amount of rotation introduced in the image.



The algorithm expects the face at a certain position in the image. If the face is rotated enough, the computer won't be able to compare image intensities properly.

3.5 Image noise

This is a particularly important concept. Imagine you wear glasses to see the blackboard while in class (as I do!). You take off your glasses to clean your eyes and notice the entire blackboard is a blur. The white chalk has become a gray blur blended into the black background of the board. The world just blurred. You can barely make of anything visually unless those glasses go back on. Well, amazingly so, the computer's mapping of intensities between corresponding pixels helps it immensely to match blurred, noisy images to their respective originals. The following images were blurred with a general filter, increasing the amount of blur from the top left (the original) clockwise to the middle left. Surprisingly, all images output the correct recognition. But nonetheless, if we blur the

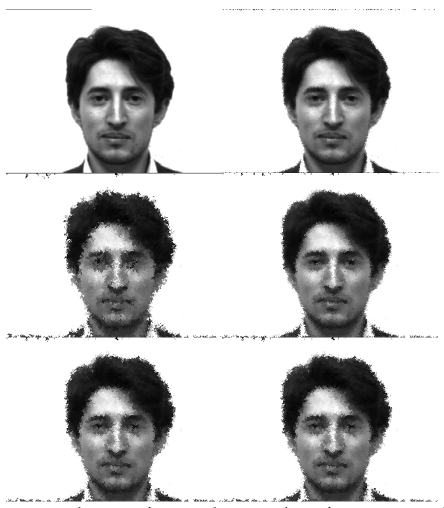
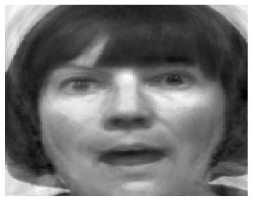
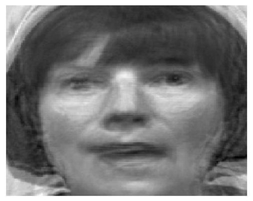


image to the point where the image and the background become one, the algorithm will fail.

3.6 Implementation of the algorithm

Now we discuss one very important implementation variation, the determination of *M'*. We tried the algorithm with *M'* = 36 and then *M'* = 100. There was a drastic change in the reconstruction of an image. The more "eigenfaces" used, the more blurred the reconstruction gets since more "eigenfaces" are included in the reconstruction. Following are two reconstructions, with *M'* = 36 on the top and *M'* = 100 on the bottom. Mathematically, we also calculated the distance between the weight representation and the distance between intensity values of corresponding pixels of the two farthest images. As we grow the number of "eigenfaces" used, the distance between the weight representations of images gets very close to the actual distance as calculated between intensity values, they almost become equal.





This is because the more "eigenfaces" used implies the more variation among faces is covered, thus the accuracy of recognizing any variant of a face increases.

4 Conclusion

While under regular "laboratory conditions" the algorithm fared quite well, after testing the algorithm under various conditions that are seen in everyday situations such as occlusion, head position, different lighting we have concluded that the algorithm is highly sensitive to these changes. A possible solution would be to increase the number of images in the database to include these changes. Also, the algorithm could adopt the Murase and Nayer approach [8] which uses a universal space to identify objects and then uses an object space to identify the object more specifically.

While face recognition is a relatively simple task for humans to perform, a robust algorithm which could perform as well as human brains has yet to be found.

5 References

[1] M.Turk and A. Pentland, “Eigenfaces for Recognition” *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[2] B. S. Manjunath, R. Chellappa, and C. von der Malsburg. “A Feature Based Approach to Face Recogntion.” In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1992.

[3] D. Valentin, H. Abdi, A. O'Toole, and G. Cotterell. “Connectionist Models of Face Processing: A Survey.” *Pattern Recognition*, 27:1209-1230, 1994.

[4] Kirby, M. and L. Sirovich, Application of the Karhunen-Loève Procedure for the Characterization of Human Faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990. **12**(1): p. 103-108.

[5] Cruise Scientific. “World of Visual Statistics: Chapter 37.” 2004. 20 April 2005. <http://www.visualstatistics.net/web%20Visual%20Statistics/Visual%20Statistics%20Multimedia/factor_analysis.htm>

[6] Lemieux, A. Parizeau, M., “Experiments on Eigenfaces Robustness” *16th International Conference on Pattern Recognition,* vol 1 pp 421-424, 2002.

[7] Kruger, J. “ Thresholds for Eigenface Recognition” 18 April 2005. <http://cnx.rice.edu/content/m12533/latest/ >

[8] H. Murase and S.K. Nayer, Visual Learning and Recognition of 3-D Objects from Appearance, *International Journal of Computer Vision*, 1995. Vol. 14, pp 5-24 .