Facial recognition with eigenfaces using the eyes region

Márcio Koch[1], Aurélio Faustino Hoppe[1], Mauro Marcelo Mattos[1]

[1]Department of Computer Science
Regional University of Blumenau (FURB) – Blumenau, SC – Brazil

lobokoch@gmail.com, {aureliof,mattos}@furb.br

Abstract. *This paper presents a facial recognition method using the eye region with eigenfaces approach and principal component analysis technique whose goal is to apply linear transformations to reduce the number of features used in the representation of the objects. The use of the most rigid part of the face, notably the eyes, reduces the problems arising from facial expressions, use of beard, mustache, lipstick, different types of hair among others. The proposed method achieved a hit rate of 99,75% using 8 eigenvectors e 5 samples per subject.*

Resumo. *Este artigo apresenta um método de reconhecimento facial que utiliza a região dos olhos, eigenfaces e análise das componentes principais cujo objetivo é aplicar transformações lineares para diminuir a quantidade de características utilizadas na representação de objetos. A partir da parte rígida do rosto, olhos, diminui-se problemas decorrentes a expressões faciais, uso de barba, bigode, batom, diferentes tipos de cabelo entre outros. O método proposto alcançou uma taxa de acerto de 99,75% usando 8 autovetores e 5 amostras por indivíduo.*

1. Introduction

According to [Tayal et al. 2013], for us humans, the most common way we recognize each other is through the face. Our brain is able to store and recognize many people through the face. The face of the people is also used to identify them in documents such as identity card, driver's license, passport, etc. There are already several applications with facial recognition, such as systems for identifying criminals and missing persons, used in public places such as airports, subways, football stadiums, busy streets and other popular use such as smartphone unlock and social networks.

Many research on eigenfaces has already been conducted in an attempt to improve facial recognition techniques. [Afolabi and Adagunodo 2012] apply an improved facial recognition algorithm in a web-based teaching system. They compare the technique of Principal Component Analysis (PCA) with Optimized Principal Component Analysis (OPCA). The result on the recognition obtained with OPCA was better than with the PCA with 98.68% and 96.60% respectively. [Shinde and Ruikar 2013] use PCA and eigenface for people recognition and obtained 97.5% accuracy using 20 main components based on Olivetti Research Laboratory (ORL) images. In the work of [Ravi and Nayeem 2013] they applied the eigenfaces approach in four different image bases and obtained the following results: Indian face database obtained 98.36% accuracy, Essex face database obtained 96.05% accuracy, Yale Database and Face 1999 obtained 93.33% and 96.15% accuracy respectively.

It can be seen from the related works that there are still challenges in facial recognition, such as: distortion of the face due to expressions such as mouth movements when speaking and smiling; Use of beard, haircut, mustache and makeup. These factors have a considerable influence on facial recognition.

To minimize the influence of factors that impair good facial recognition accuracy, this study deals with facial recognition using only the most rigid face region, which comprises the area just from above the eyebrows to below the eyes, along with the technique of eigenfaces with PCA.

2. Face Recognition Approach

According to [Loesch and Hoeltgebaum 2012], PCA aims to transform a set of correlated variables into a smaller set of uncorrelated variables called principal components. The first major component contains the greatest possible variability, while each successive component contains part of the remaining variability. For this, the data covariance matrix is decomposed by its corresponding eigenvectors and eigenvalues.

The facial recognition algorithm proposed in this work can be divided into three main phases: pre-processing, eigenfaces calculation and recognition.

In the preprocessing stage, we try to adjust (transformations) the images in a way that enhances the feature extraction algorithm and, at the same time, reduces the dimensionality of the information processed. The main pre-processing steps are: (a) detection and normalization of the face pose, (b) extraction of the eye region and (c) resizing of the images. During de step (a) the face and eyes are detected using a cascade classification algorithm, also known as "Haar-like" [Bradski and Kaehler 2001]. This technique is based on the detector approach proposed by [Viola and Jones 2004], using the AdaBoost learning algorithm to organize cascade rejection nodes for each region of the analyzed image.

After detecting the face, we need to find the eyes. They can be found also using [Viola and Jones 2004] approach. If necessary, the face is rotated by interpolating the pixels so that the eyes are on the same horizontal line. To do this, we need to find the center of the eyes and verify if they are aligned horizontally by calculating the angles between them. If the angle of inclination is positive, the face is rotated clockwise, if it is negative, it is rotated counterclockwise, as shown in Figure 1.

(a) (b)

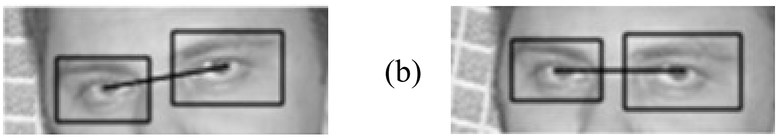

Figure 1. Example of normalization of face pose. (A) horizontally misaligned face with eye line. (B) face aligned.

To extract only the eye region step (b) it is necessary to obtain the coordinates of the rectangle formed by the face according to the Haar algorithm. Thus, a region of interest is created (region just above the eyebrows below the eyes) with the coordinates presented according the Equation 1, where x is the coordinate of the upper left corner, y is the upper corner coordinate, w is the width of the rectangle and h is the height of the rectangle. Figure 2 shows an example of the result of this operation.

$$x = 0, y = \frac{1}{8}.h, x = \ face\ width, h = \frac{y}{2} \quad (1)$$

Figure 2. Example of region of interest

In step (c), the images are resized using pixel interpolation so that they all have the same resolution as 64 x 64 pixels. This aims to balance computational cost without losing the important information contained in the image. Figure 3 shows an example.

Figure 3. Example of a face resized to 64 X 64 pixels.

At this point the PCA with eigenfaces techniques are used to extract only the main characteristics that describe each image of the face and stores them in a smaller set of images, called eigenfaces, that represent in a reduced form the set of all the training images used.

For the formation of patterns with images of the face we use the technique where a face image with height h and width w can be represented as an array of pixels containing h lines and w columns [Campos 2001]. To transform a face image into a vector of patterns, one can create a vector x of size n, where n = h*w.

Each pixel of the image is a feature that will be part of the vector. To create this vector the image is read column by column. It can be said that an image is a point in an n-dimensional space, called an image space. Thus, an image of 64 x 64 pixels will be represented by a point in a space of 4,096 dimensions.

Given a set of M training faces transformed into pattern vectors and added as columns in a pattern matrix Γ [nx M] = (Γ_1, Γ_2, ..., Γ_M), the mean face can be calculated from Equation 2 [Turk and Pentland 1991].

$$\Psi_{[n\,x\,1]} = \frac{1}{M}\sum_{i=1}^{M}\Gamma_{i[n\,x\,1]} \quad (2)$$

In Figure 4 it is possible to visualize the generation of the average face of four face images. They are not represented in their pattern vector, but in their original state.

Figure 4. Generation of the mean face.

The difference between each face and the mean face is stored in vector Φ_i and is calculated by the Equation 3, since A [n x M] = (Φ_1, Φ_2, ..., Φ_M) is the matrix that will store the differences between each face and the mean face.

$$\Phi_{i[n\,x\,1]} = \Gamma_{i[n\,x\,1]} - \Psi_{[n\,x\,1]} \quad (3)$$

The covariance matrix of the vector population Φ_i can be calculated from the Equation 4, where the elements C_{ii} of the matrix are the variance of x_i, the i-th component of the vector x of the set, and the element C_{ij} is the covariance between the elements x_i and x_j of these vectors. When the elements x_i and x_j are not correlated, their covariance is zero, so $x_i = x_j = 0$.

$$C_{ij[nxn]} = \frac{1}{M}\sum_{i=1}^{M} \Phi_{i[n\,x\,1]}\Phi^{T}_{i[1\,x\,n]} = A_{[n\,x\,M]}A^{T}_{[M\,x\,n]} \quad (4)$$

According to [Gonzalez and Woods 2011], as the covariance matrix C is real and symmetric it is possible to find a set of n orthonormal eigenvectors. However, calculating the eigenvectors and eigenvalues of an array of order n x n (N^2) for the typical size of images for facial recognition becomes computationally intractable. A simplified way of calculating eigenvectors and eigenvalues is by the Equation 5.

$$L_{[M\,x\,M]} = A^{T}_{[M\,x\,n]}A_{[n\,x\,M]} \quad (5)$$

Usually the set of faces for training is much smaller ($M \ll N^2$) than the number of pixels contained in each image. Thus, the first eigenvectors of C can be expressed as a linear combination between the eigenvectors of L (denoted by V) and the vectors (faces images) contained in A.

The eigenfaces are calculated as follows: Let V_i and λ_i, for i = 1, 2, ..., n, eigenvectors and eigenvalues respectively of L ordered in decreasing order such that $\lambda_i \geq \lambda_i + 1$, for i = 1 , 2, ..., n-1. The first line of V has the eigenvector corresponding to the largest eigenvalue and the last line of V has the eigenvector corresponding to the lowest eigenvalue.

To maximize the variance of the characteristics and to cancel out the covariance between the characteristics of the patterns generated from the images of the faces, it is necessary to diagonalize the covariance matrix. For this it is necessary to diagonalize the covariance matrix. For this it is necessary to apply a transform known as the Hotelling transform, which maps each pattern x into a new vector y according to the Equation 6.

$$U_{[n\,x\,M]} = A_{[n\,x\,M]} V_{[M\,x\,M]} \tag{6}$$

The matrix U has an eigenface (eigenvector) in each column. If we display each of these vectors as an image, they resemble the faces used for training, but with the main features highlighted as shown in Figure 5. It is possible to notice that the first five eigenfaces have a higher contrast intensity, this is because they have the highest eigenvalues associated. It is also possible to state that the variance is a measure of contrast intensity.

(a)

(b)

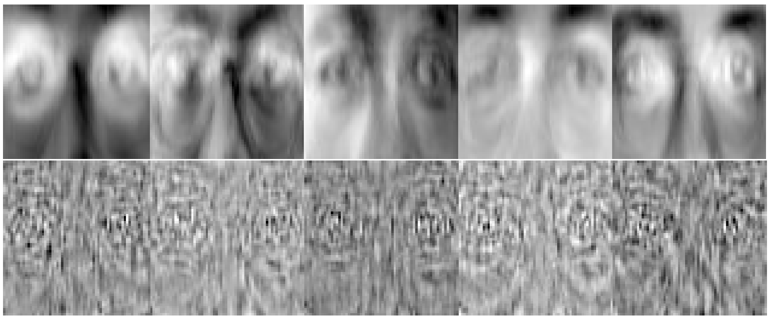

Figure 5. Eigenfaces displayed as images. (a) the first five eigenfaces. (b) the last five eigenfaces.

Since the first k eigenfaces are sufficient to represent the main characteristics of the faces, it is not necessary to use all of them for recognition, which considerably reduces the dimensionality of the information processed. A face Γ can be projected as a point on a k-dimensional image space through a vector ω containing the weights calculated through the Equation 7. It is possible to see that the dimension k to project the face as a point in the space of images is much smaller than the size of the original N^2 face.

$$\omega_{[k\,x\,1]} = U^{T}_{[k\,x\,n]}(\Gamma_{[n\,x\,1]} - \Psi_{[n\,x\,1]}) \tag{7}$$

Figure 6 shows the weights and their respective projections in the image subspace for a face of size 64 x 64 pixels with k = 10. Without using the PCA, 4,096 dimensions would be required to project the same image.

	*k*	1	2	3	4	5	6	7	8	9	10
	Weight	1939	153	354	246	-905	321	-154	-266	237	-103

Figure 6. One-sided weights calculated to effect their projection in the subspace of images.

The projections of the training faces should be calculated using the Equation 7 and stored in each line of the matrix = (ω_1, ω_2, ..., ω_M) T which describes the contribution of each eigenface in the representation of the faces. Analogous to the training faces, a test face is calculated and the weights are stored in the vector.

2.1. Recognition phase

The following describes the steps for recognizing an unknown test face. It is possible to reconstruct a face by multiplying the eigenfaces by the face projection weights, added to the middle face, as shown in the Equation 8. The more eigenfaces are used, the more accurate the face reconstruction will be, but the greater the computational cost to perform the PCA calculations.

$$\Gamma_{f[n\,x\,1]} = U_{[n\,x\,k]}\Omega_{[k\,x\,1]} + \Psi_{[n\,x\,1]} \quad (8)$$

The reconstruction error between the face and its reconstruction is calculated by the Equation 9.

$$\varepsilon = \left\|\Gamma_{[n\,x\,1]} - \Gamma_{f[n\,x\,1]}\right\| \quad (9)$$

The Equation 10 shows how to calculate the vector modulus and thus obtain the Euclidean distance between the vector of the test face and each of the training faces stored in the database. Using the nearest K-NN technique, we use K = 1, is stored at the shortest distance between the test face and the training faces, including the index of the nearest face.

$$\varepsilon_k = \min(\left\|\Omega_{[Mx1]} - \Omega_{[Mxk]}\right\|), for\ k = 1, 2, \ldots, M \quad (10)$$

The following are the steps of the algorithm for recognition and identification of a test face Γ: (a) calculate the projection weights of the test face and the training faces through the Equation 7 and store them in the vectors respectively; (b) reconstruct the test face using the Equation 8; (c) calculate the reconstruction error and store it in ε using the Equation 9; (d) calculate which of the training faces is closest to the test face and store in ε_k according to the Equation 10; (e) define a maximum threshold θ_ε for the shortest distance between the test face and the training faces; (f) set a maximum threshold Θ_ε for the reconstruction error of the test face; and (g) perform the recognition of the test face following the three steps algorithm:

- If ($\varepsilon_k < \theta_\varepsilon$ AND $\varepsilon < \Theta_\varepsilon$) then the test face is recognized and corresponds to the individual of the training face with index k;
- If ($\varepsilon k >= \theta_\varepsilon$ AND $\varepsilon < \Theta_\varepsilon$) then the test face is a human face, but it is not in the database, so it is unknown;
- If ($\varepsilon >= \Theta_\varepsilon$) then the test face may not be a human face.

3. Experiments and Results

Following the results of the experiments are presented using facial recognition techniques presented in this paper. The results were measured by the accuracy rate in the recognition of individuals.

The faces images used in the experiments of this article were made available by [AT&T 2002] through the ORL base. This dataset has 40 individuals, each with 10 different faces images with size 92 x 112 pixels and varying illumination, facial expressions, glasses, beard and mustache, as showed in Figure 7.

Figure 7. Some samples from ORL database.

In the first experiment the number of images was varied, using one face of each individual for tests, 10 eigenvectors, maximum threshold for distance 1,700,000 and maximum reconstruction threshold 30. These values were reached through several attempts until reaching the recognition rate. The results of the experiment are shown in Table 1.

Table 1. Recognition with varying amount of images for training

Number of images	2	3	4	5	6	7
Recognition (%)	57.50	70.00	85.00	99.75	99.75	99.75

As observed, when the amount of training images increases, the recognition rate also improves. From the fifth image of training, the maximum rate of recognition of 99.75% was reached, and from this value the rate remained the same. However, increasing the number of training images does not improve recognition, while that will take up more disk space and will require more processing in training and recognition operations.

Finding an ideal number of training images can be a hard task to be done. In this experiment the number of images is small, where with five training samples image we already reach the ideal rate of recognition, but in a large database, can be necessary to use many more samples images for training. We use a trial and error technique to find the ideal number of training samples and it seems that is what most literature also uses.

In the second experiment, the number of images per subject was fixed, being 5 for training and one for testing. The number of eigenvectors, the maximum distance threshold and the reconstruction threshold were varied, as shown in Table 2.

From Table 2 the distance threshold and the reconstruction threshold used are inversely proportional as the eigenvectors are increased. The distance increases with the increase of the dimensionality of the characteristics used, and the maximum would be the number of pixels of the images used, which in our case is 4,096 with images of 64 x 64 pixels.

Table 2. Experimental data with variation of eigenvectors, distance threshold, reconstruction threshold and percentage of recognition

Eigenvectors	Distance threshold	Reconstruction threshold	Recognition (%)
5	700.000	30.0	87.50
6	1.300.000	35.0	90.00
7	1.400.000	34.0	97.50
8	1.400.000	33.0	99.75
12	2.200.000	29.0	99.75
20	2.900.000	27.4	99.75
50	4.902.000	24.0	97.50
80	5.901.000	21.7	97.50

The reconstruction threshold decreases due to the increase of eigenvectors with higher eigenvalues used. According to [Gonzalez and Woods 2011], using all eigenvectors in the reconstruction of the original image from the eigenfaces, it will have reconstruction error close to zero. However, it is not computationally feasible to use all eigenvectors. Very good recognition rates can be achieved by using only the eigenvectors that point to the main directions.

Table 2 also shows that by adding new characteristics (using more eigenvectors) the recognition rate increases. With 8 eigenvectors the maximum recognition rate is reached. From this value, up to 20 eigenvectors are stable. Adding more characteristics, the recognition rate is impaired due to the excess of characteristics.

Some comparative results among our method and correlated works can be seen in Table 3. Our method using only eye region characteristics can reach a good recognition rate with eight eigenfaces.

Table 3. Comparative results by recognition methods

Recognition method	Number of eigenfaces	Detection rate
[Shinde and Ruikar 2013]	20	97.50%
[Ravi and Nayeem 2013]	-	98.36%
[Afolabi and Adagunodo 2012]	-	98.68%
Our method	8	99.75%

4. Conclusion and Future Work

One of the great advantages of this approach is the tolerance to changes in certain regions of the face without compromising recognition, for example: use of beard and mustache, haircut, use of cosmetics, change in facial expression, etc.

However, to achieve a good hit rate (above 90%), several training images per individual are required (in this work we use at least 5 images), which is not always possible in real applications. Alignment of faces is also required, and the pose must be frontal, tolerating little inclination.

The computational time and cost for face training is proportional to the number of faces and number of eigenfaces used, the longer the processing time. Recognition is quick compared to the training phase.

From the results obtained we are starting a new phase in the tests using larger image bases, with more than 1,000 faces images. In addition, we will be working to improve tolerance to variation of lighting, scale and pose of faces.

References

Afolabi, Adeolu, O.; Adagunodo, Rotimi. Implementation of an improved facial recognition algorithm in a web based learning system. International Journal of Engineering and Technology, Ogbomoso, v. 2, n. 11, p. 1-8, Dez. 2012.

AT&T. The database of faces. Cambridge, 2002. Disponível em: <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>. Acesso em: 01 out. 2018.

Bradski, Gary; Kaehler, Adrian. Learning OpenCV. Sebastopol: O'Reilly books, 2008. Campos, Teófilo E. Técnicas de seleção de características com aplicação em reconhecimento de faces. 2001. 160 f. Dissertação (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo.

Campos, Teófilo E. Técnicas de seleção de características com aplicação em reconhecimento de faces. 2001. 160 f. Dissertação (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo.

Gonzalez, Rafael C.; Woods, Richard E. Processamento digital de imagens. 3. ed. São Paulo: Pearson, 2011.

Loesch, Cláudio; Hoeltgebaum, Marianne. Métodos estatísticos multivariados. São Paulo: Saraiva, 2012.

Ravi Subban; Nayeem Sadique. A study on face recognition technique based on eigenface. International Journal of Applied Information Systems, New York, v. 5, n. 4, p. 57-62, Mar. 2013.

Shinde, Anagha A.; Ruikar, Sachin D. Face recognition using PCA and eigen face approach. International Journal of Computer Applications, New York, v. 3, n. 3, p. 7-12, Mai. 2013.

Tayal, Yogesh; Pandey, Pramod, K.; Singh, D. B. V. Face recognizion usin eigenface. International Journal of Emerging Technologies in Computational and Applied Sciences, Gwalior, v. 3, n. 1, p. 50-55, Dez. 2013.

Turk, Matthew; Pentland, Alex. Eigenfaces for recognition. Journal of Cognitive Neuroscience, Cambridge, v. 3, n. 1, p. 71-86, Jan. 1991.

Viola, Paul; Jones, Michael, J. Robust real-time face detection. International Journal of Computer Vision, [S.l.], v. 57, n. 2, p. 137-154, Mai. 2004.