Genetic Search for Face Detection and Verification

George Bebis†, Satishkumar Uthiram† and Michael Georgiopoulos‡

†Department of Computer Science, University of Nevada, Reno
‡Department of Electrical and Computer Engineering, University of Central Florida
{bebis, uthirams}@cs.unr.edu, mng@ece.engr.ucf.edu

Abstract

We investigate the application of Genetic Algorithms (GAs) to search for the face of a particular individual in a two-dimensional intensity image. This problem has many potential applications such as locating a person in a crowd from pictures taken by surveillance cameras. There are two steps in solving this problem: first, faces regions must be extracted from the image (*face detection*) and second, candidate faces must be compared against the face of interest (*face verification*). Without any *a-priori* knowledge about location and size of a face in an image, every possible image location and face size must be considered, leading to a very large search space. In addition, face detection or verification invariant to lighting conditions, facial expression and pose make the search space even larger and more complex. In this paper, we propose using GAs to search the image efficiently. Specifically, we use GAs to find image sub-windows that contain the face of interest. Each sub-window is evaluated using a fitness function which contains two terms: the first term favors sub-windows containing faces while the second term favors sun-windows containing faces similar to the face of interest. Both terms have been defined using ideas from the method of "eigenfaces". A set of increasingly complex scenes demonstrate the performance of the genetic search approach.

Keywords: face detection, face verification, genetic algorithms, eigenfaces.

1. Introduction

In this paper, we consider the problem of searching for the face of a particular individual in a two-dimensional image. This problem has many potential applications such as locating a person in a crowd from pictures taken by surveillance cameras. There are two main sub-problems behind this problem: *face detection* and *face verification*. Specifically, given an image, the first step is to extract all possible regions that might contain a face. The second step is to compare the extracted faces against the face of interest. Both of these problems are very challenging. Without any *a-priori* knowledge about the location and size of a face in an image, we will have to consider almost every possible location and size, leading to a very large search space. In addition, changes in lighting conditions, facial expression and pose make search space even larger and more complex.

For face verification, methods based on correlation [13] and neural networks [14] are quite common. For face detection, many methods consider color [1][2] (e.g., skin color distribution) or motion [3][4] information to find the face region(s) quickly without resorting to an exhaustive search. This information, however, might not always be available. In general, methods on face detection can be classified into two main categories: methods based on facial features [7][8] or face models [5][6] (e.g., template) and methods based on face representations learned from a large number of examples (face images) using statistical approaches (e.g., eigenfaces) [9][10] or neural networks [11][12]. In general, methods in the second category are more practical since they are less time consuming and do not rely on special features or models.

The method of eigenfaces uses Principal Components Analysis (PCA) to linearly project the image space to a low dimensional subspace (eigenspace). This subspace is defined by the principal components (eigenfaces) of the distribution of face images (i.e., the most important eigenvectors of the covariance matrix of the set of faces). Each face can be represented as a linear combination of the eigenfaces. Given an image, sub-images of different size are extracted at every image location. To classify an image as a face, its distance from the eigenspace space is computed. In [11][12], retinally connected neural networks examine sub-windows of the image to decide whether they contain a face. Extensive training using a representative set of both face and non-face examples is required for the neural networks to learn the concept of face.

In this paper, we investigate the idea of using Genetic Algorithms (GAs) [15][16] to search for the face of interest. GAs are search procedures which have shown to perform quite well when the search space is very large. This is the case with the problem of searching for the face of a particular individual in a complex scene. GAs operate iteratively on a population of structures,

each one of which represents a candidate solution to the problem at hand. Each structure, is modified in a much the same way that populations of individuals evolve under natural selection. Variations among the individuals in the population result in some individuals being more fit than others (i.e., better solutions). In the past, GAs have been used to solve various difficult problems such as target recognition [17], facial feature extraction [18], and object recognition [19].

In this work, the GA starts by extracting random sub-windows from the input image. These sub-windows are retained in subsequent generations if they contain a face, especially if it is the face that we are looking for. To evaluate a sub-window, we use a fitness function which contains two terms. The first term favors sub-windows which contain faces while the second term favors sub-windows which contain faces similar to the face of interest. Both terms have been defined using ideas from the method of eigenfaces [10]. In the original eigenface approach, a single eigenspace is employed, built from a large set of images from different individuals. Both face detection and recognition are then performed using the same eigenspace (i.e., by computing the "distance from face space" and "distance in face space"). Here, we use two different eigenspaces.

The first eigenspace is built using images from different individuals, as in the original approach, and is used to define the first term of the fitness function (face detection term). There is a difference, however, between the original approach and the one used here: we do not use the original gray-scale images but the images obtained after applying the Sobel operator [20]. Our experiments have indicated that this preprocessing step enhances facial features (e.g., face contour, eyes, mouth, nose) making it easier to distinguish between faces and non-faces. The second eigenspace is built using different images of the face of interest obtained under some variation in lighting conditions, facial expression, and pose. The second term of the fitness function (face verification term) is defined using the distance from this eigenspace. It should be mentioned that in a practical application, we might not be able to obtain images of the face of interest under different conditions. In this case, somebody might try to generate synthetic images from a small number of real images [25].

The approach proposed here has similarities with the approach of Swets and Punch [21]. In their approach, the GA extracts sub-windows from the image as well, however, the sub-windows may fall outside the input image. To deal with this problem, they included an extra term to the fitness function to penalize those windows. This, however, adds unnecessary complexity to the fitness function and increases time. Here, we use a smart encoding scheme to make sure that all the sub-windows extracted by the GA fall inside the input image. Also, they evaluate the extracted sub-windows by computing the distance of the sub-window from the mean of the face of interest. Here, we employ a more powerful fitness function based on the method of eigenfaces.

The rest of the paper is organized as follows: In Section 2, we provide brief review of the eigenface approach. Section 3 presents the proposed method in detail while Section 5 presents our experimental results. Finally, Section 6 contains our conclusions and discussion.

2. The method of eigenfaces

The method of eigenfaces is based on Principal Component Analysis (PCA), a standard statistical technique for reducing the dimensionality of data while attempting to preserve as much of information as possible in terms of variance. The key idea is to represent each data in a low dimensional space defined by the most important eigenvectors (i.e., "eigenfaces") of the covariance matrix of the data distribution. A complete description of the eigenface approach can be found in [10]. Here, we just summarize the main ideas.

Representing each image $I(x, y)$ as a $N \times N$ vector Γ_i, first the average face Ψ is computed: $\Psi = \frac{1}{R}\sum_{i=1}^{R}\Gamma_i$, where R is the number of faces in the training set. Next, the difference Φ of each face from the average face is computed: $\Phi_i = \Gamma_i - \Psi$. Then, the covariance matrix is estimated by:

$$C = \frac{1}{R}\sum_{i=1}^{R}\Phi_i\Phi_i^T = AA^T \qquad (1)$$

where, $A = \left[\Phi_1\ \Phi_2 \ldots \Phi_R\right]$. The eigenspace can then be defined by computing the eigenvectors u_i of C. Since C is very large ($N^2 \times N^2$), computing its eigenvectors will be very expensive. Instead, we can compute v_i, the eigenvectors of A^TA, an $R \times R$ matrix. Then, u_i can be computed from v_i as follows (the details are given in [10]):

$$u_i = \sum_{j=1}^{R} v_{ij}\Phi_j, \qquad j = 1, \ldots, R \qquad (2)$$

Usually, we only need to keep a smaller number of eigenvectors R', corresponding to the largest eigenvalues. Given a new image Γ, we subtract the mean ($\Phi = \Gamma - \Psi$) and we compute its projection: $\hat{\Phi} = \sum_{i=1}^{R'} w_i u_i$, where $w_i = u_i^T\Phi$ are the coefficients of projection.

An image is considered to be a face if the mean square error (called the *distance from face space (dffs)*) between its representation using the most important eigenvectors and its normalized counterpart (e.g., the difference of the input image and the mean image), is small. Also, an image is considered to be a face found in the data set if the error (called the *distance within face space*

(difs)) between the coefficients of the eigenvectors used to represent the image and the face in the data set is small.

3. Methodology

In this section, we describe the genetic search approach. Figure 1 shows the main steps. The following sections discuss the steps in detail.

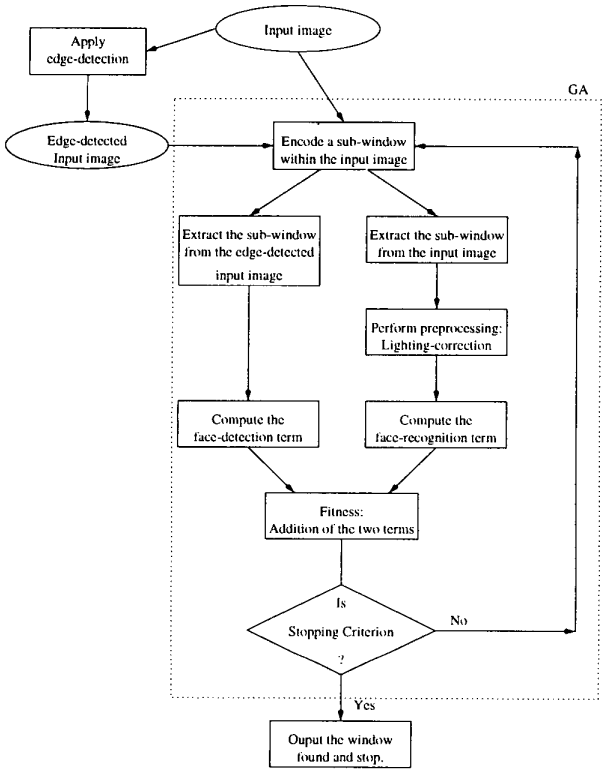

Figure 1. The main steps of the proposed approach.

3.1. Preprocessing

To compute the eigenfaces, first the training images must be registered. The procedure used is similar to the one given in [11]. Specifically, the eyes, tip of the nose, and the corners and center of the mouth of each face were labeled manually. These points were then used to normalize each face to same scale, orientation and position. The normalization was performed by mapping the facial features to some fixed locations in an N x M image. The mapping was assumed to be an affine transformation, computed iteratively as in [11]. Figure 2 shows some examples of images before and after normalization.

Each normalized image was then preprocessed to account for different lighting conditions and contrast. First, a linear function was fit to the intensity of the image. The result was subtracted out from the original image to correct lighting differences. Then, histogram equalization [20] was performed to correct for different camera gains and to improve contrast. Figure 3 shows some examples.

Figure 2. Example showing images before and after normalization.

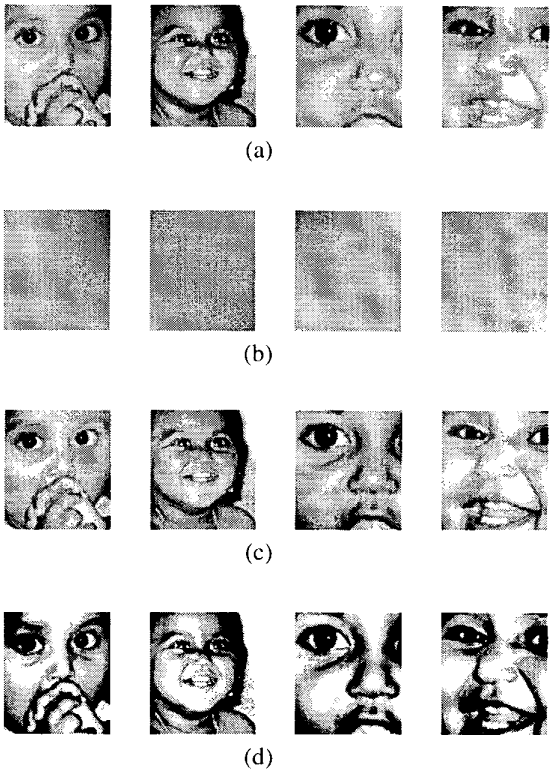

Figure 3. The preprocessing steps: (a) original image, (b) linear fit, (c) light corrected image, (d) histogram equalized image.

3.2. Eigenface representation

All the images in the training set were preprocessed as explained in the previous paragraph. Then, they were used to compute the eigenspace representation. As we have already discussed, we are using two different eigenspaces: the first eigenspace is used for face detection and is built using images of different individuals. To

improve face detection, we enhance the facial features of each face by computing the image gradient using the Sobel operator [20]. A 3 x 3 mask was used in our experiments. The resulted images were then thresholded (i.e., if a value was below 30, it was set to zero, otherwise, it remained unchanged) and used to built the first eigenspace. Figure 4 shows sample images and some of the computed eigenfaces. To built the second eigenspace, we used different images from the individual of interest. To allow for some robustness, the images were obtained by allowing some variation in lighting conditions, facial expression, and pose.

Figure 4. The first row shows the enhanced faces while the second row shows the mean and the first four eigenfaces.

3.3. Encoding

In our encoding scheme, each individual (chromosome) in the population represents a sub-window within the given input image. There are some constraints that need attention when encoding a rectangular window. First of all, to evaluate sub-windows of different size using the eigenface approach, we need to scale them down or up to the size of the images in the training set. Although we tried several different scaling techniques [20], we decided to use nearest neighborhood interpolation [20] since it is fast and gives satisfactory results for our application. Second, to avoid face distortions, we need to make sure that the sub-windows chosen by the GA have the same aspect ratio with that of the images in the training set [21]. While trying to maintain the aspect ratio, we also need to make sure that each chromosome defines a window that lies within the bounds of the input image. Finally, through our experiments we found that the eigenface approach does not work very well when using small sub-windows. Specifically, we ended up with a large number of false positives when the size of the sub-window was very small. Thus, we constrained the GA choose sub-windows not smaller than $MinX$ x $MinY$ ($MinX = 50$ and $MinY = 48$). These conditions are depicted in Figure 5.

To define a sub-window, while at the same time satisfy the constraints discussed above, we encode three of the four points that define a window. We employed two different versions of encoding as shown in Figure 6. The selection of Scheme1 is based upon the condition that the aspect ratio of the input image is greater than the aspect ratio of the images in the training set. If the opposite is true, Scheme2 is used (the reason in discussed below). UL stands for the upper-left corner of the sub-window, and BR stands for the bottom-right corner of the defined sub-window. Let L be the length of each chromosome. Then, L is given by $L = 3 * m$, where m is the number of bits used to represent each point encoded in the chromosome. The following constraint should be satisfied:

2^m >= max(input image's height, input image's width)

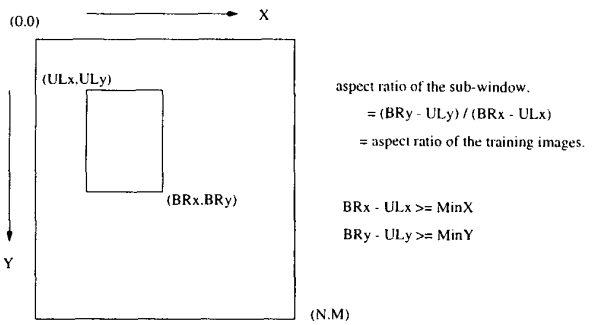

Figure 5. The constraints that must hold true for a sub-window.

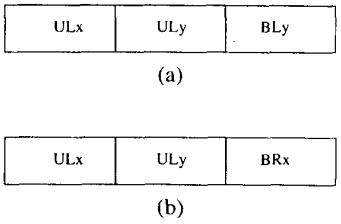

Figure 6. Our encoding schemes, (a) scheme1, (b) scheme2.

Given that the sub-window lies within the image and that its aspect ratio is maintained, we need to compute the fourth point that completes the definition of the boundaries of the sub-window. The following discussion is based on Figure 5 and on the assumption that we use encoding scheme1 (Figure 6(a)). Let A be the aspect ratio of the images in the training set. In order for the sub-window and the training images to have the same aspect ratio, the fourth point, BR_y, that completes the definition of the window, is calculated as follows:

$$BR_y = A * (BR_x - UL_x) + UL_y \quad (3)$$

BR_y should satisfy the following inequality:

$$MinY \leq BR_y < M \quad (4)$$

Substituting (3) in (4), we have:

$$MinY \leq A * (BR_x - UL_x) + UL_y < M \quad (5)$$

or

$$c \leq BR_x < d \quad (6)$$

where $c = (MinY - UL_y + A * UL_x)/A$ and $d = (M - UL_y + A * UL_x)/A$. BR_x should also lie inside the image, thus, it should satisfy the following inequality:

$$0 \leq BR_x < N - MinX \quad (7)$$

To decode BR_x, a linear mapping from $[0, 2^m - 1]$ to $[max(0, c), min(N - MinX, d)]$ is used. The other two points, UL_x and UL_y, are decoded using a linear mapping from $[0, 2^m - 1]$ to $[0, N - MinX]$ and from $[0, 2^m - 1]$ to $[0, M - MinY]$, respectively. Using (3) we can compute the value of BR_y. Similar calculations are performed when using encoding scheme2. To see when each scheme needs to be used, let us consider (6). From (6), we have that $MinX \leq c$ and $d < N$. Let us assume for simplicity that $MinX = MinY = 0$. Then, $0 \leq 0c$ implies that $A < M/N$ while $d < N$ implies that $UL_x < N$ which is always true. Thus, if $A < M/N$ scheme1 is used, otherwise, scheme2 is used.

3.4. Fitness Evaluation

Each individual in the population needs to be evaluated. Based on its fitness it will be decided if it will survive in subsequent generations. As we have already discussed, the fitness function used here contains two terms. The first term denotes how close the sub-window is to the face space (face detection term). The second term, denotes how close the sub-window resembles the face of interest (face verification term). To compute the face detection term, we compute its $dffs_{detection}$ using the eigenspace built from different individuals. To compute the face verification term, we compute its $dffs_{verification}$ using the eigenspace built from different face images from the individual of interest. This has yielded better results than including the images of the face of interest in the first eigenspace, using the "distance in face space" as the second term of the fitness function.

The less the value of $dffs_{detection}$, the more the sub-window resembles a face; and the less the value of $dffs_{verification}$, the more it resembles the face of interest. Thus, both of these values provide a measure of error. Since we need to maximize the fitness but minimize the error, our fitness function is given as:

$$Fitness = MAX - dffs_{detection} - dffs_{verification}$$

which changes the minimization problem to a maximization problem for the GA (MAX is a large constant value).

4. Experimental Results

We have used two training sets of faces in our experiments. The first set includes 38 images (see Figure 7(b) and is used to compute the face detection term of the fitness function. As we have already discussed, we do not use the original images for building the eigenspace but the images obtained after applying the Sobel operator, followed by thresholding (see Figure 7(c)). The second set contains 20 face images from the individual of interest. Figure 7(a) shows one of the sets we used in our experiments. Different lighting, facial expressions and slight tilts were allowed to make verification more robust.

Figure 7. (a) lighting-corrected, histogram-equalized training set used for face verification, (b) original images used for face detection (c) images processed with the Sobel operator, followed by thresholding.

Our selection strategy was cross generational. Assuming a population of size P, the offspring double the size of the population and we select the best P individuals from the combined parent-offspring population for further processing [17]. This kind of selection does

well with small populations and leads to quick convergence (sometimes prematurely). We also linearly scale fitnesses to try maintain a constant selection pressure. A simple two-point crossover and point mutation were employed in our experiments. The crossover probability used was 0.95, the mutation probability was 0.05 and the scaling factor was 1.2. The *MAX* constant in the fitness function was set to 18000.

We have tested our algorithm on several scenes. Here, we show results on seven scenes of increasing complexity (Scene1 - Scene7) with the face of interest being present in all of them (see Figure 8). The population size was set to 100 for Scene1 and Scene2 and to 150 for all other scenes. On the average, 40 generations were required for the algorithm to find the face of interest. For each scene, we tested our approach 10 times with different random seeds. Performance plots (not shown here due to lack of space) indicate that the GA gets close to the correct solution quickly and then spends most of its time making little progress.

Figure 8(a). Scene1 (Size: 120 *x* 128)

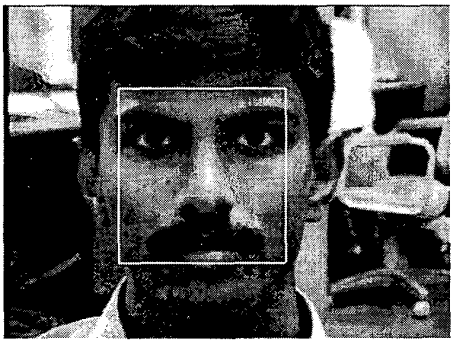

Figure 8(b). Scene2 (Size: 240 *x* 320)

Scene1 is relatively simple containing only the face of interest. What is interesting about this image is that it is not a very recent picture of the person of interest. The GA was able to find the face in all cases (see Figure 8(a)). Next, we tested GA's ability to find the face of interest, assuming different size. Figure 8(b) shows Scene2 with the face of interest being very close to the camera. The GA converged to the face in all experiments. To test GA's robustness against occlusion, we tested it on Scene3 where the person of interest is wearing glasses. As seen in Figure 8(c), the GA performed a good job finding the face. It should be mentioned that in all three cases, the fintness of the solutions found was very high. We have also tested images containing different faces. The GA converged to these faces as well, however, the fitness of the solutions was much lower.

Scenes 4-7 were used to test GA's performance in the presence of other faces. In the case of Scenes 4 and 5, the faces present resemble the face of interest as they are from the same ethnic group and all have moustache. In all ten experiments, the face of interest was found correctly. Scenes 6 and 7 contain faces from other races. In the case of Scene 6, the GA converged to the face of interest 6 out of 10 times. The other two times it converged to the face shown in Figure 8(f). Our analysis indicated that the two solutions have close fitnesses. In the case of Scene 7, the GA converged to the face of of interest 9 out of 10 times. The face to which it converged one time is shown in Figure 8(g). Our analysis indicated again that the fitnesses were close (but not as close as in the case of Scene 6). These two cases indicate that more powerful fitness functions are required. We elaborate on this issue in the next section. Nevertheless, we were able to achieve 100% accuracy by increasing the population size.

Figure 8(c). Scene3 (Size: 350 *x* 444)

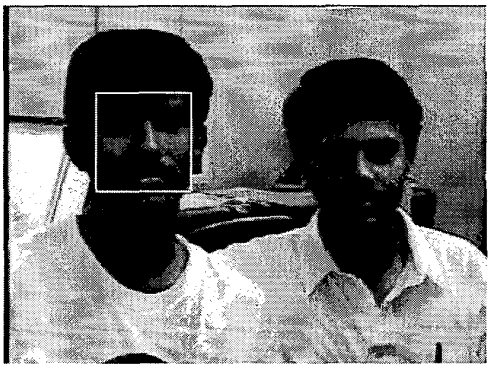

Figure 8(d). Scene4 (Size: 240 *x* 320)

Figure 8(e). Scene5 (Size: 269 x 395)

Figure 8(f). Scene6 (Size: 438 x 497)

Figure 8(g). Scene7 (Size: 288 x 464)

Figure 8. Some of the scenes considered in our experiments. The faces found are shown in a white box.

The total number of sub-windows that the GA explores is much less compared to the entire search space. To compute this number, we consider the product of population size and number of generations it took for the GA to converge to the correct face. To estimate the efficiency gained, we computed the number of all possible sub-windows assuming that the smallest sub-window is of size 50 x 48 (since the GA does not consider smaller sizes than this). Table 1 shows the ratio of sub-windows searched by the GA over the total number of possible sub-windows. Clearly, the GA was able to find the face of interest by searching only a very small portion of the solution space.

Table 1. *Summary of results.*

Results				
Scene	*Image Size*	*Problem Size*	$GA_{matches}$	*Ratio*
1	*120x128*	*8,051,400*	*3200*	*0.000397*
2	*240x320*	*673,688,000*	*4000*	*0.00000594*
3	*350x444*	*3,549,060,000*	*6975*	*0.00000197*
4	*240x320*	*673,688,000*	*14625*	*0.0000217*
5	*269x395*	*1,454,510,000*	*11250*	*0.0000077*
6	*438x497*	*7,623,950,000*	*17100*	*0.0000022*
7	*288x464*	*2,466,860,000*	*21150*	*0.00000857*

5. Conclusions

In this paper, we have proposed using genetic algorithms to search for the face of a particular individual in an image. Specifically, we used GAs to search for sub-images that might contain the face of interest. To evaluate each sub-window, we proposed a fitness function using ideas from the eigenface approach. Also, we proposed an encoding scheme which ensures that the sub-images extracted lie inside the input image and have the same aspect ratio with the training images. Our experimental results demonstrate the GAs is a promising tool for solving this problem.

The current work has certain limitations. First of all, we consider mostly frontal face images. Second, the lighting conditions do not change very much. Finally, facial expression and pose do not vary considerably. Actually, these limitations are not limitations of the GA approach itself but limitations of the eigenface-based fitness function. We were able to allow for some degree of tolerance by employing a separate eigenspace for the face of interest, however, more powerful approaches need to be used in defining the fitness function. Among the methods we plan to investigate in the future are the method of Fisherfaces [23] and the method of Independent Component Analysis (ICA) [24].

References

[1] G. Yang and A. Waibel, "A Real-time Face Tracker", *Workshop on Applications of Computer Vision*, pp. 142-147, 1996.

[2] H. Wu, Q. Chen, and M. Yachida, "Face Detection from Color Images using a Fuzzy Pattern Matching Method", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 557-563, 1999.

[3] C. Wang and M. Brandstein, "A Hybrid Real-Time Face Tracking System," *Proceedings of ICASSP*, pp. 3737-3740, 1998.

[4] A. Eleftheriadis and A. Jacquin, "Automatic face location detection for model-assisted rate control in H.261-compatible coding of video", *Signal Processing: Image Communication*, vol. 7, no. 4-6, pp. 435-455, 1995.

[5] S. Birchfield, "An elliptical head tracker", *31st Asilomar Conference*, pp. 1710-1714, 1997.

[6] Y. Kwon and N. da Vitoria Lobo, "Face detection using templates", *Computer Vision and Pattern Recognition Conference*, pp. 764-767, 1994.

[7] C. Huang and C. Chen, "Human Facial Feature Extraction for Face Interpretation and Recognition", *Pattern Recognition*, vol. 25, no. 12, pp. 1435-1444, 1992.

[8] R. Brunelli and T. Poggio, "Face Recognition: Features versus Templates", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 10, pp. 1042-1052, 1993.

[9] M. Kirby and L. Sirovich, "Application of the Karhunen-Loe've Procedure for the Characterization of Human Faces",*IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 1, pp. 103-108, 1990.

[10] M. Turk and A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, Vol. 3, pp. 71-86, 1991.

[11] H. Rowley, S. Baluja and T. Kanade, "Neural Network-Based Face Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, pp. 23-38, 1998.

[12] K. Sung and T. Poggio, "Example-based Learning for View-based Human Face Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 1, pp. 39-51, 1998.

[13] J. Matas, K. Jonsson, and J. Kittler, "Fast face localisation and verification", *Image and Vision Computing*, vol. 17, pp. 575-581, 1999.

[14] R. Winter, "Verification of Personal Identity using Facial Images", *SPIE Conference*, vol. 2277, pp. 63-70, 1994.

[15] J. Holland, *Adaptation in natural and artificial systems*, The Universi ty of Michigan Press, 1975, Ann Arbor.

[16] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[17] A. Katz and P. Thrift, "Generating Image Filters for Target Recognition by Genetic Learning", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 9, pp. 906-910, 1994.

[18] C. Lin and J. Wu, "Automatic Facial Feature Extraction by Genetic Algorithms", *IEEE Transactions on Image Processing*, vol. 8, no. 6, pp. 834-844, 1999.

[19] G. Bebis, S. Louis, and Y. Varol, "Using Genetic Algorithms for Model-based Object Recognition", *International Conference on Imaging Science, Systems, and Technology (CISST'98)*, Las Vegas, pp. 1-6, 1998.

[20] R. Crane, *A simplified approach to Image Processing*, Prentice Hall, 1997.

[21] D. Swets, B. Punch and J. Weng, "Genetic Algorithms for Object Localization in a Complex Scene", *International Conference on Image Processing*, vol II, pp. 595-598, 1995.

[22] L. Eshelman, "The CHC adaptive search algorithm: How to have safe search when engaging in non-traditional genetic recombination", In Gregory J. E. Rawlins (ed.), *Proceedings of the Foundations of Genetic Algorithms Workshop - 1*, SanMateo, CA, 1990.

[23] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using Class Specific Linear Projection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 711-720, 1997.

[24] M. Bartlett and T. Sejnowski, "Independent components of face images: A representation for face recognition", *4th Joint Symposium on Neural Computation Proceedings*, Pasadena, CA, May 1997.

[25] D. Beymer and T. Poggio, "Face Recognition from One Example View", *International Conference on Computer Vision*, pp. 500-507, 1995.