An Eye Detection System Based on Neural Autoassociators

Monica Bianchini and Lorenzo Sarti

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Siena
Via Roma 56 — 53100 Siena, Italy

Abstract. Automatic eye tracking is a challenging task, with numerous applications in biometrics, security, intelligent human–computer interfaces, and driver's sleepiness detection systems. Eye localization and extraction is, therefore, the first step to the solution of such problems. In this paper, we present a new method, based on neural autoassociators, to solve the problem of detecting eyes from a facial image. A subset of the AR Database, collecting individuals both with or without glasses and with open or closed eyes, has been used for experiments and benchmarking. Preliminary experimental results are very promising and demonstrate the efficiency of the proposed eye localization system.

1 Introduction

Human face detection is often the first step in numerous applications, such as video surveillance, human–computer interface [1], face recognition, and image database management. Moreover, facial feature extraction, especially with frontal images, has a wide range of usage in automated face modelling, facial expression recognition, face animation, feature–based face recognition, and driver's sleepiness detection [2]. The problem of detecting human eyes has attracted a considerable interest in computer vision society. Many efforts have been addressed to capture the essential physical and emotional information from eyes. In intelligent vehicle systems, eye gaze and the motion of eye pupil provide important information for fatigue analysis [3]. In face detection and recognition systems, eyes can provide the richest identity information [4].

Many different approaches are reported in literature to address the problem of eye detection, based on some observations that could be made on the peculiarities of the "object" to be detected. For instance, since the pupils generally appear darker w.r.t. the surrounding regions, some algorithms search for local gray minima [5]. Techniques such as contrast enhancement and intensity thresholding are involved, in order to extract the dark regions. In [6], the eye regions are located based on an a priori knowledge on the facial feature arrangement (the hair region has the largest area in the binary image, the eyes are situated below the eyebrows, etc.). However, such algorithms are highly sensitive to the thresholding method used, as well as to the lighting conditions, i.e. the gray level information would be helpful in detecting several eye candidates, but it

F. Schwenker and S. Marinai (Eds.): ANNPR 2006, LNAI 4087, pp. 244–252, 2006.

may not be sufficient to filter out different facial features, such as eyebrows, which also appear as dark patches. Alternatively, an artificial template could be built, according to the rough shape of the eye and the eyebrow, such that the correlation coefficient between the template and the eye image can be calculated [7,8]. Hough transform was also employed [9], which implies a preliminary robust edge detection procedure. Finally, more recently, attention has been payed to Gabor wavelets techniques [10], where Gabor wavelet–based linear filters are used for eye corner detection, and non–linear (Gaussian) filters are used for iris detection. All the above mentioned methods belong to the class of feature–based approaches, whereas image–based techniques, like Principal Component Analysis (PCA), have also been applied [11]. Despite these efforts, robust, accurate, and non–intrusive eye detection and tracking remains largely an unsolved issue. The challenges result from eye closure, eye occlusion, variability in scale and face orientation, and different lighting conditions.

In this manuscript, when we refer to the eyes, we are considering not only the iris, but rather the collection of contours forming the pupil, iris, eyelids, eyelashes, eyebrows and the shading around the eye orbit. This general eye region is a larger and more dominant structure as a whole than its individual subcomponents. Therefore, it is more stable and easier to detect. Although the process of including the surrounding region improves robustness, it reduces accuracy since the contours of the eyebrows and eye orbit shading may have a center that does not coincide with the pupil's center.

The method proposed consists of three fundamental steps. First, a preprocessing phase, based on the application of the Sobel filters, is carried out on color images, in order to extract the principal contours [12]; then observing that the eyes exhibit strong transitions, because of the iris and the white part of the eye, the projections of horizontal and vertical transitions are evaluated [13]. Finally, both the projections are used to train two specialized neural autoassociators.

The paper is organized as follows. In the following section, the feature extraction method is described, whereas Section 3 briefly sketches some salient properties of the neural autoassociators used for detecting the eyes. In Section 4, preliminary but promising results are reported, whereas Section 5 collects some conclusions.

2 Feature Extraction

The proposed eye detection technique is based on neural autoassociators and on gradient features extracted from the images. The eyes possess strong horizontal and vertical edges [14], therefore the exploitation of gradient features is particularly suited to represent the image content.

In order to determine the gradient transitions, both the Sobel operators [12] are applied to the input image to determine vertical and horizontal edges. Then, the horizontal and vertical projections are computed summing the rows and the columns of the filtered image, respectively (see Fig. 1).

In [13], a thresholding technique is proposed to analyze the projections and to localize the eyes. Unfortunately, this method can fail when the image presents

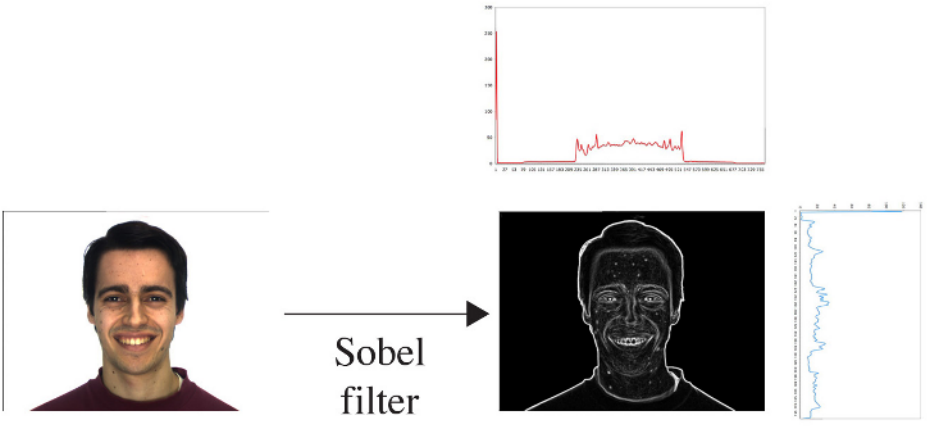

Fig. 1. The image is processed using the Sobel operators, then the horizontal and vertical projections are extracted

sharp gradient transitions, for instance, for open mouth faces or in presence of glasses. To overcome such problems, the method proposed in this paper employs two neural autoassociators (one for each projection). For the sake of simplicity, the procedure will be described w.r.t. the horizontal projections, since, as a matter of fact, the vertical projections are managed in the same way.

The horizontal projections are scanned from the top to the bottom of the input image, using a moving window. Initially, the window position corresponds to the top of the image and then it moves down pixel by pixel. For each window position, a vector of integer features, that collects both the window position and the values of the projections that lie behind the window, is created. Therefore, each input vector is an $n + 2$ array of integers, $\mathbf{x} = (a, b, x_1, x_2, \ldots, x_n)$, where n is the window dimension in rows, a and b represent the indexes of the rows that delimit the window, and each x_i counts the number of white pixels in the i–th row inside the window. In order to train the neural autoassociator, a target, that assesses if the window position corresponds to the eye area or to a part of it, is associated to each feature vector. A target equal to 1 corresponds to a feature vector that represents a part of the eye area, while a value equal to 0 is associated to the feature vectors that do not belong to the eye area, i.e. if the window intersects the eye area but it is not completely included, then the associated target is posed to 0. Obviously, the window width must be smaller than the eye area, in order to have a set of vectors corresponding to such area. Moreover, the target association is performed knowing the position of the eyes in the training images (the associated ground–truth information is needed).

In order to train the neural autoassociators, a set of training images must be chosen. For each image, two sets of feature vectors, corresponding to horizontal and vertical projections, are extracted and the relative targets are associated. Thus, the autoassociator which deals with the horizontal projections is

specialized to locate the vertical position of the eyes, while the other one performs the horizontal localization.

An eye localization system can exploit the trained autoassociators as follows:

- The system is fed with an input image, on which the Sobel operators are applied to compute the gradient transitions, and hence to determine the horizontal and vertical projections;
- Using the moving window technique, two sets of feature vectors are extracted, considering both the horizontal and the vertical projections;
- Each neural autoassociator processes its set of feature vectors and predicts, for each vector, if the associated window position corresponds to a part of the eye area;
- Finally, adjacent window positions predicted as eye locations are merged together to determine the bounding boxes that correspond to the eyes.

The localization system is able to determine the correct position of the eyes if the input and the training images meet the following constraints. First, we need to process facial images, depicted in foreground. Then, persons must appear in a frontal view and only a small inclination of each face is allowed. Finally, each image must possess an uniform background. Nevertheless, the last constraint can be overcome by integrating the system with a preprocessing module for face localization. In fact, face localization allows to reduce the noise represented by the background. Many methods proposed in the past exploit skin–color filters to localize faces. As a matter of fact, the human skin colors range in a relative small region, independently of the particular color space chosen to represent the images [15]. Unfortunately, the performances of skin–color filters deeply depend on the light conditions and on the ethnic group of the depicted persons. In order to overcome such limitations, appearance–based methods, able to infer face models using machine learning techniques, are preferable [16,17].

3 Eye Detection Using Autoassociators

Autoassociators are a special kind of neural networks which, by learning to reproduce a given set of patterns, grasp the underlying concept that is useful for pattern classification. The number of inputs and outputs in autoassociators corresponds to the dimension of the input space, whereas a smaller number of units forms the hidden layer. Each autoassociator is trained to reconstruct an input $\mathbf{x}$ at the output $\mathbf{t}$, and its parameters are optimized to minimize the Euclidean distance $||\mathbf{x} - \mathbf{t}||^2$. To achieve an accurate reconstruction, the autoassociator is implicitly forced to discover an appropriate nonlinear mapping of the original input space into a smaller space that captures the properties of the underlying distribution.

Autoassociators are generally used as one–class learning machines. In other words, each network corresponds to a particular category and, during training, it receives only the samples within the category. An important consequence is that the network will learn to accurately reproduce positive samples (those in the corresponding class), producing a prototype for that class. Thus, autoassociators

provide an alternative approach to concept learning. In particular, the higher the reproduction quality for an input pattern, the more likely it belongs to the category for which the autoassociator is constructed. Moreover, the specialization of each autoassociator to a particular class may be reinforced by training each network also on negative examples and forcing the prototype to be as far as possible from patterns outside the proper class.

In this paper we use two autoassociators, one for each set of projections, with sigmoid hidden neurons and linear output units. In [18], such architectures are proved to realize a sort of clustering in the input space. Moreover, an end–of–learning condition was stated, assessing that, at the end of the learning process, an equality relation holds between the output correlation matrix $\mathbf{X}_2'\mathbf{X}_2$ and the input/output correlation matrix $\mathbf{X}_2'\mathbf{X}_0$, with $\mathbf{X}_0$ and $\mathbf{X}_2$ collecting (by row) all the training examples and the related outputs, respectively. From the geometrical point of view, such an equality may be interpreted as a loss of energy spent in the association process, which is higher when the autoassociator is not able to perfectly reproduce the presented target at the output (i.e. the hidden compressed representation is not sufficient to hold all the information needed to reconstruct the pattern).

The standard Backpropagation algorithm, with adaptive learning rate, was used for training, based on two different error functions related to patterns belonging or not to the class represented by the autoassociator. In fact, for positive patterns the quadratic error is minimized, whereas the error function for negative examples is based on the inverse distance:

$$E_p = \sum_{t \in C} ||\mathbf{X}_2(t) - \mathbf{X}_0(t)||^2,$$

$$E_n = \sum_{t \notin C} \frac{1}{||\mathbf{X}_2(t) - \mathbf{X}_0(t)||^2 + \varepsilon},$$

where ε, which can be chosen proportional to the machine precision, was introduced for guaranteeing numerical stability. The connection weights are then

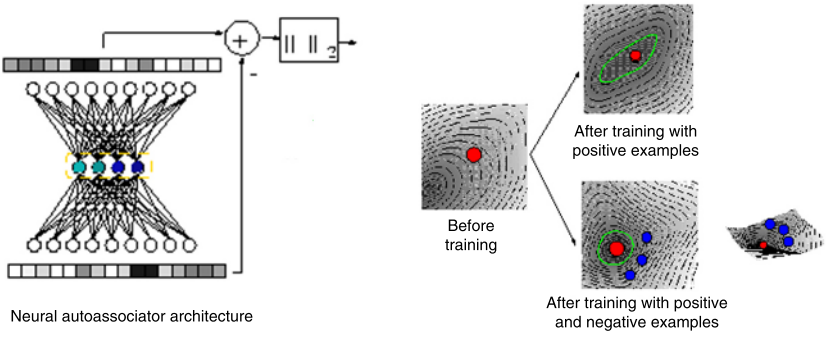

Fig. 2. Changes on the error surface due to training with positive or positive/negative patterns

updated based on the joint contributions of E_p and E_n. The effects of the introduction of negative examples in the training set consist in focusing the autoassociator on the class is devoted to represent, so that it creates a prototype for that class which is as far as possible from examples belonging to other classes (see Fig. 2). Finally, an *ad hoc* threshold must be chosen to establish whether each pattern belongs or not to a particular class, i.e. is properly represented by the prototype produced by the corresponding autoassociator. Such a threshold could assume different values for different classes, due to the distribution of the input patterns, and is generally computed via a trial–and–error procedure.

4 Experimental Results

In order to evaluate the effectiveness of the proposed technique, some experiments were carried out using a subset of the AR Database [19]. This dataset collects 4000 color images corresponding to 126 individuals (70 men and 56 women). The images represent frontal view faces with different facial expressions, illumination conditions, and occlusions, obtained with sunglasses or scarfs (see Figure 3).

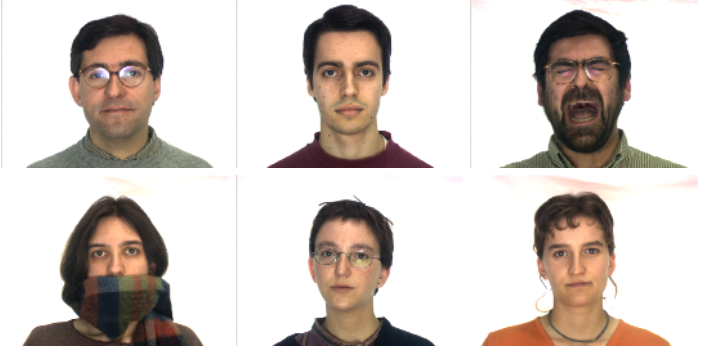

Fig. 3. Samples of images from the AR database

Our subset collects 210 images randomly chosen from the original database (faces occluded by sunglasses were excluded). The images were subsequently divided into two sets that collect the same number of images: the training and the test sets. Each image was represented as described in Section 2. The reported results are referred to a window width equal to 10 rows/columns of pixels, for horizontal and vertical projections. However, the performances of the system are not particularly affected by this parameter when the window dimension is smaller than the average eye width (height). For each image, 566 vectors represent the horizontal projections, and 758 vectors describe the vertical projections. The percentage of negative examples (vectors extracted using a window position that corresponds to the eye location) is equal, on average, to 2.47% and to

Table 1. Equal error rates obtained by the localization system, varying the number of hidden units. The classification thresholds are reported in brackets.

Architecture	Horizontal projections	Vertical projections
6 hidden	90.02% (0.135)	91.04 (0.68) %
7 hidden	90.85% (0.21)	91.84% (0.73)
8 hidden	90.05% (0.115)	88.23% (0.735)

Table 2. Effects of the classification threshold choice. The results were obtained using a neural autoassociator with 7 hidden units.

Horizontal projections			
Classification threshold	Non eye area Accuracy	Eye area Accuracy	Global Accuracy
0.2	90.54%	91.22%	90.55%
0.15	88.91%	94.62%	89.05%
0.1	86.42%	97.48%	86.7%
Vertical projections			
Classification threshold	Non eye area Accuracy	Eye area Accuracy	Global Accuracy
0.65	90.32%	92.32%	90.55%
0.6	89.15%	93.84%	89.7%
0.55	87.93%	94.99%	88.75%

11.61% considering the horizontal and vertical projections, respectively. Several training runs were performed to determine the best autoassociator architectures. The obtained results are summarized in the following tables. Table 1 reports the equal error rates of the neural autoassociators, varying the number of hidden units. The equal error rate is defined as the accuracy of the system when the number of errors in the two classes is equal. Practically, the sensitivity of the system can be chosen varying the classification threshold. When decreasing the classification thresholds, the neural autoassociators increase their ability to localize the eyes. In Table 2, the accuracy rates obtained varying the classification thresholds are reported, showing that the decrease of such parameters does not drastically deteriorate the performances of the whole system, whereas the percentage of negative examples correctly autoassociated (i.e. the percentage of eye localizations) grows significantly. In fact, in eye detection systems, a high recall is generally preferable w.r.t. a high precision and, thus, the choice of a threshold smaller than that able to obtain the equal error rate should be advisable.

5 Conclusions

In this paper, an eye localization system is proposed, based on autoassociators, which are trained on horizontal and vertical projections obtained by color images

after the application of the Sobel operators. The preliminary experimentation, carried out on a subset of the AR Database, shows very promising results, allowing a best global accuracy of 91.84% for vertical projections, with a recall on the negative examples (i.e. those identifying the eye area) of 95%, whereas 90.85% and 97.48% are the best accuracy and recall for horizontal localization, respectively. It is a matter of future work varying the dimension of the window used for scanning the images, both horizontally and vertically, and trying to collect more informative features (like the area and/or an approximation of the shape of the projections) into the vectors used to train the autoassociators.

References

1. D. Ward and D. McKay, "Fast hands–free writing by gaze direction," *Nature*, vol. 418, no. 6900, p. 838, 2002.
2. M. Eriksson and N. Papanikotopoulos, "Eye tracking for detection of driver fatigue," in *Proceedings of IEEE Int. Conf. on Intelligent Transportation Systems*, pp. 314–319, IEEE, 1997.
3. Q. Ji and X. Yang, "Real–time eye, gaze, and face pose tracking for monitoring driver vigilance," *Real–Time Imaging*, vol. 8, no. 5, pp. 357–377, 2002.
4. R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 696–706, 2002.
5. M. Rizon and T. Kawaguchi, "Automatic eye detection using intensity and edge information," in *Proceedings IEEE TENCON*, vol. 2, pp. 415–420, IEEE, 2000.
6. L. Zhang and P. Lenders, "Knowledge–based eye detection for human face recognition," in *Proceedings of the 4^{th} Int. Conf. on Knowledge–based Intelligent Systems & Allied Technologies*, pp. 117–120, IEEE, 2000.
7. M. Betke and W. Mullay, "Preliminary investigation of real–time monitoring of a driver in city traffic," in *Proceedings of IEEE Intelligent Vehicles Symposium*, IEEE, 2000.
8. S. A. Suandi, S. Enokida, and T. Ejima, "An extended template matching technique for tracking eyes and mouth in real–time," in *Proceedings of Visualization, Imaging and Image Processing*, pp. 586–591, 2003.
9. T. Kawaguchi, D. Hidaka, and M. Rizon, "Detection of eyes from human faces by Hough transform and separability filter," in *Proceedings of Int. Conf. on Image Processing*, pp. 49–52, 2000.
10. S. Sirohey and A. Rosenfeld, "Eye detection in a face image using linear and nonlinear filters," *Pattern Recognition*, vol. 34, pp. 1367–1391, 2001.
11. M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proceedings IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 586–591, IEEE, 1991.
12. R. Gonzalez and R. Woods, *Digital Image Processing.* Addison Wesley, 1992.
13. X. Deng, C.-H. Chang, and E. Brandle, "A new method for eye extraction from facial image," in *Proceedings of IEEE DELTA*, IEEE, 2004.
14. D. Maio and D. Maltoni, "Real–time face location on gray–scale static images," *Pattern Recognition*, vol. 33, pp. 1525–1539, 2000.
15. J. Yang, W. Lu, and A. Waibel, "Skin–color modeling and adaptation," in *Proceedings of ACCV'98*, vol. 2, pp. 687–694, 1998.
16. M. Bianchini, M. Maggini, L. Sarti, and F. Scarselli, "Recursive neural networks learn to localize faces," *Pattern Recognition Letters*, vol. 26, pp. 1885–1895, 2005.

17. A. Carleson, C. Cumby, J. Rosen, and D. Roth, "The SNoW learning architecture," Tech. Rep. UIUCDCS–R–99–2101, University of Illinois at Urbana–Campaign Computer Science Department, 1999.
18. M. Bianchini, P. Frasconi, and M. Gori, "Learning in multilayered networks used as autoassociators," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 512–515, 1995.
19. A. Martinez and R. Benavente, "The AR face database," Tech. Rep. 24, CVC Technical Report, 1998.