Defence Science Journal, Vol. 61, No. 5, September 2011, pp. 431-442, DOI: 10.14429/dsj.61.1178

Face Recognition using Segmental Euclidean Distance

Farrukh Sayeed[*], M. Hanmandlu[#], and A.Q. Ansari[!]

[*]*PA College of Engineering, Mangalore*
[#] *Indian Institute of Technology Delhi, New Delhi*
[!]*Jamia Millia Islamia, New Delhi*
E-mail: sayeed.farrukh@gmail.com

ABSTRACT

In this paper an attempt has been made to detect the face using the combination of integral image along with the cascade structured classifier which is built using Adaboost learning algorithm. The detected faces are then passed through a filtering process for discarding the non face regions. They are individually split up into five segments consisting of forehead, eyes, nose, mouth and chin. Each segment is considered as a separate image and Eigenface also called principal component analysis (PCA) features of each segment is computed. The faces having a slight pose are also aligned for proper segmentation. The test image is also segmented similarly and its PCA features are found. The segmental Euclidean distance classifier is used for matching the test image with the stored one. The success rate comes out to be 88 per cent on the CG database created from the databases of California Institute and Georgia Institute. However the performance of this approach on ORL database with the same features is only 70 per cent. For the sake of comparison, discrete cosine transform (DCT) and fuzzy features are tried on CG and ORL databases but using a well known classifier, support vector machine (SVM). Results of recognition rate with DCT features on SVM classifier are increased by 3 per cent over those due to PCA features and Euclidean distance classifier on the CG database. The results of recognition are improved to 96 per cent with fuzzy features on ORL database with SVM.

Keywords: Face Segmentation, principal component analysis, DCT and fuzzy features, segmental euclidean distance, SVM

1. INTRODUCTION

The face recognition is one of the most important applications of the Biometric-based authentication systems. During the past few decades, it has received a considerable attention, because of a large number of application areas such as entertainment, smart cards, information security, law enforcement, surveillance, etc. Face recognition involves a range of activities from many aspects of human life. Humans are good at face recognition and now machine learning is being improved to do this task. Identification of people in different environments requires three steps: acquisition, normalisation, and recognition. Acquisition involves the detection and tracking of face-like patches in the dynamic scenes. Normalisation is the segmentation, alignment and standardisation of the face images, and finally recognition is concerned with the representation via modelling of face images as identities, and association of face images with the known models.

The methods reported in the literature are broadly classified into:

(a) Holistic matching method in which the whole face acts as an input to the recognition system,
(b) Feature-based matching method, in which as the name indicates, the local features such as eyes, mouth, and nose are first extracted and their locations and the local statistics are fed to the recognition system and
(c) Hybrid method where the recognition system makes use of both the local features as well as the whole face region to recognise the face.

1.1 Literature Survey on Face Detection

Schneiderman and Kanade[8] apply statistical likelihood tests, using feature output histograms to create the detection scheme. Rowley and Kanade[7] use neural network-based filters to obtain the benchmark results. In another early work, Papageorgiou[5], *et al.* proposes a general object detection scheme which uses a wavelet representation and statistical learning techniques. Osuna[4], *et al.* apply Vapnik's support vector machine for the face detection, and Romdhani[6], *et al.* improve on that work by creating the reduced training vector sets. Fleuret and Geman attempt a coarse-to-fine approach to face detection, focusing on minimising the computation[3]. In perhaps the most impressive paper, Viola and Jones use the concept of an integral image, along with the rectangular feature representation and Adaboost learning algorithm to detect faces at the rate of 15 frames per second[2].

We now present an overview of some human face recognition techniques suitable to the frontal faces. The methods considered are eigenfaces, neural networks, dynamic link architecture, hidden Markov model and geometrical

Received 12 February 2011, published online 2 September 2011

feature matching. These approaches are analysed in terms of the facial representations adopted by them.

1.1.1. Eigenfaces

Eigenface also known as Karhunen-Loève expansion or the principal components analysis (PCA) is one of the most extensively investigated representations of a face[20-21]. Any face image can be approximately reconstructed by a set of weights for each face and a standard eigenpicture. The weights describing each face are obtained by projecting the face image onto the eigenpicture by the technique of Kirby and Sirovich[21].

1.1.2. Neural Networks

The attractiveness of neural networks lies in the easiness with which non linearity can be introduced in the network model. The earliest artificial neural network (ANN) used for the face recognition is a single layer adaptive network called WISARD and such a separate network is required to store the model of a person[11]. Other networks for the face detection include multilayer perceptron[12] and convolution neural network[13]. A multi-resolution pyramid structure is employed for the face recognition. A hybrid neural network combines the local image sampling, a self-organising map (SOM) neural network and a convolutional neural network.

1.1.3. Graph Matching

Graph matching is another approach for the face recognition. An extension of ANN called dynamic link structure is presented[15] for the distortion invariant object recognition that employs an elastic graph matching to find the closest stored graph. The stored objects are represented by sparse graphs, whose vertices are labelled with a multiresolution description in terms of a local power spectrum and whose edges are labelled with geometrical distance vectors. Object recognition is formulated as an elastic graph matching performed by the stochastic optimisation of a matching cost function.

1.1.4. Hidden Markov Models (HMMs)

Hidden Markov models (HMMs) are proved to be successful for the stochastic modelling of non stationary vector time series like speech, but these are applied to the human face recognition[16]. Faces are intuitively divided into regions such as the eyes, nose, mouth, etc. which in turn can be associated with the states of a hidden Markov model. Since HMMs require a one-dimensional observation sequence whereas the images are two-dimensional, the images should be converted into either 1D temporal or spatial sequences.

1.1.5. Geometrical Feature Matching

Geometrical feature matching techniques rely on a set of geometrical features derived from the face image. The fact that the face recognition is possible even at the coarse resolution as low as 8 x 6 pixels[17] at which the facial features are hardly revealed in detail implies that the overall geometrical configuration of the face features is sufficient for the recognition. The overall configuration can be described by a vector comprising the position and size of the main facial features, such as eyes and eyebrows, nose, mouth, and the shape of the face outline.

2. DETECTION OF A FACE

The face detector is used to extract regions of an input image containing only the faces along with small area of the background. The detector uses the face detection method based on the Viola-Jones OpenCV Face-Detection[2] using Mex. The main reason for this implementation is that it is feature based and therefore relatively fast compared with the model-based detector. This approach is now elaborated.

2.1. Viola-Jones Open CV Face-Detection

A brief description of the feature Algorithm due to Viola Jones[2] contains three parts: (a) Integral Image, (b) Ada Boost learning algorithm, and (c) Cascade Structured Classifier.

2.1.1. Integral Image

Wavelet analysis is a tool for signal processing. Here we use it to focus on the localised area of the image with a view to find whether it contains a face. A one dimensional Haar wavelet function is just a step function as shown in Fig 1(a) and we need two-dimensional wavelet functions shown in Fig 1(b) which do not satisfy some conditions of the wavelets.

It may be noted that Haar wavelet features are amenable to be computed efficiently using an integral image denoted by $II(x; y)$ of image $I(x; y)$, which is defined as

$$II(x_2, y_2) = \sum_{x'<x, y<y} I(x'y') \quad (1)$$

The integral image $II(x, y)$ can be computed in one pass over the original image starting with $S(x, 0) = 0$ and $II(0, y) = 0$ by using a pair of recurrence relations:

$$S(x, y) = S(x, y-1) + I(x, y)$$
$$II(x, y) = II(x-1, y) + S(x, y) \quad (2)$$

where $S(x, y)$ is the cumulative sum of the x^{th} row. Any rectangular sum in an image can be expressed in terms of its integral image as illustrated in Fig. 2. The sum of the pixels within a rectangle D in Fig. 2 can

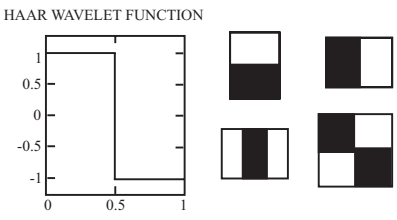


Figure 1. (a) 1-D Haar wavelet function, (b) 4 types of 2-D Haar wavelet functions.

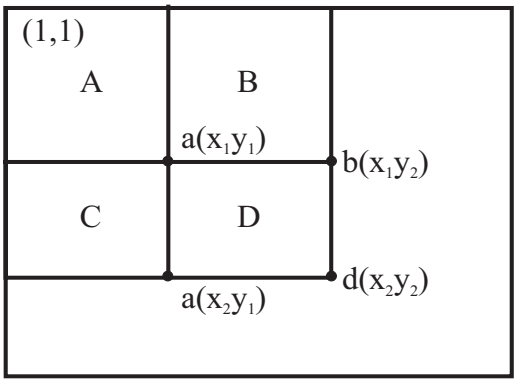


Figure 2. The rectangular sum using an integral image.

be computed using the values of the points *a, b, c, d* in the integral image as follows:

$$II(x_2, y_2) = \sum_{x<x_2, y<y_2} I(x, y) = A + B + C + D$$

$$II(x_2, y_1) = \sum_{x<x_2, y<y_1} I(x, y) = A + C$$

$$II(x_1, y_2) = \sum_{x<x_1, y<y_2} I(x, y) = A + B$$

$$II(x_1, y_1) = \sum_{x<x_1, y<y_1} I(x, y) = A$$

$$D = II(x_2, y_2) + II(x_1, y_1) - II(x_2, y_1) - II(x_1, y_2) \qquad (3)$$

The use of the integral images leads to enormous savings and makes it possible to use Haar wavelets in a real-time detection system.

2.1.2. AdaBoost Learning Algorithm

In our biometric system a variant of AdaBoost is used both to select the features and to train the classifier. In its original form, the AdaBoost learning algorithm is used to boost the classification performance of a simple learning algorithm (e.g., it can be used to boost the performance of a simple perceptron). It does this by combining a collection of weak classification functions to form a stronger classifier. In the language of boosting the simple learning algorithm is called a weak learner. So, for example the perceptron learning algorithm searches over the set of possible perceptrons and returns the perceptron with the lowest classification error. The learner is called weak because we do not expect even the best classification function to classify the training data well (i.e. for a given problem the best perceptron may only classify the training data correctly 51% of the time). In order for the weak learner to be boosted, it is called upon to solve a sequence of learning problems. After the first round of learning, the examples are re-weighted in order to emphasize those which were incorrectly classified by the previous weak classifier. The final strong classifier takes the form of a perceptron, a weighted combination of weak classifiers followed by a threshold.

The formal guarantees provided by the AdaBoost learning procedure are quite strong. Freund and Schapire proved that the training error of the strong classifier approaches zero exponentially over the number of rounds. More importantly several results prove the generalization performance of AdaBoost. The key insight is that generalization performance is related to the margin of the examples, and that AdaBoost achieves large margins rapidly.

The conventional AdaBoost procedure can be easily interpreted as a greedy feature selection process. Consider the general problem of boosting, in which a large set of classification functions are combined using a weighted majority vote. The challenge is to associate a large weight with each good classification function and a smaller weight with poor functions. AdaBoost is an aggressive mechanism for selecting a small set of good classification functions which nevertheless have significant variety. Drawing an analogy between weak classifiers and features, AdaBoost is an effective procedure for searching out a small number of good 'features' which nevertheless have significant variety.

The core inference provided in the system is brought through the use of weak classifiers where a weak classifier hj is a simple structure containing a feature vector f_j, threshold θ_j and parity p_j. Here the output of the classifier is dependent on whether or not the feature value is less than the threshold, hj. A weak classifier set is essentially produced when the complete feature set has been derived. From the feature set it is possible to evaluate the individual classifier thresholds (across a large image dataset) and also define the parity of each classifier.

The base resolution of the detector is 24 x 24, which results in a large number of possible features. A small number of these features can be combined to form an effective classifier. A variant of AdaBoost is used to select a small set of features and train the Cascade Classifier. The operation of the AdaBoost learning algorithm is illustrated in Fig. 3.

For the task of face detection, the initial rectangle features selected by AdaBoost are meaningful and easily interpreted. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. This feature is relatively large in comparison with the detection sub-window, and should be somewhat insensitive to size and location of the face. The second feature selected relies on the property that the eyes are darker than the bridge of the nose.

In summary the 200-feature classifier provides initial evidence that a boosted classifier constructed from rectangle features is an effective technique for object detection. In terms of computation, this classifier is probably faster than any other published system. Unfortunately, the most straightforward technique for improving detection

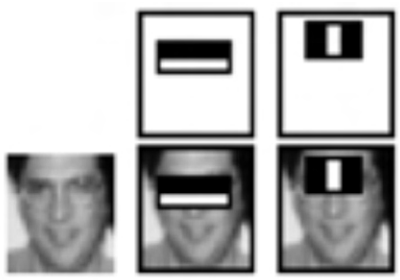


Figure 3. The first and second features selected by AdaBoost. The two features are shown in the top row and then overlaid on a typical training face in the bottom row.

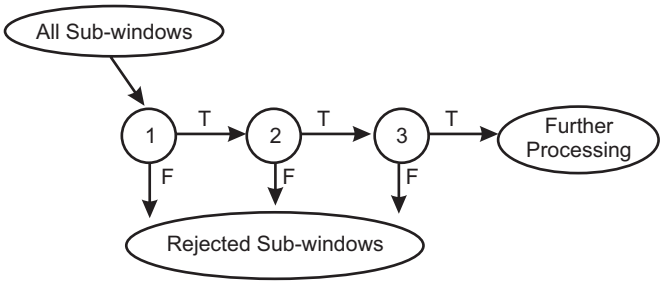


Figure 4. Schematic depiction of a detection cascade.

performance, adding features to the classifier, directly increases computation time.

2.1.3. Cascade Structured Classifier

It is important to create a cascade structured classifier that rejects many of the negative sub-windows while detecting almost all positive instances. A more efficient boosted classifier rejects many of the negative sub-windows while detecting almost all positive instances (i.e. the threshold of a boosted classifier can be adjusted so that the false negative rate is close to zero). Simpler classifiers are used to reject the majority of the sub-windows, before more complex classifiers are called upon to achieve low false positive rates. The overall detection process yields a degenerate decision tree, called 'cascade' (See Fig. 4). A positive result from the first classifier triggers the second classifier to achieve very high detection rates. A positive result from the second classifier triggers a third classifier, and so on. Occurrence of a negative outcome at any point leads to the immediate rejection of the subsequent sub-window.

Stages in the cascade are constructed by training the classifiers using AdaBoost and the threshold is adjusted to minimise the false negatives. Note that the default AdaBoost threshold is designed to yield a low error rate on the training data. In general a lower threshold yields higher detection rates with higher false positive rates.

A series of classifiers is applied to every sub-window. The initial classifiers are selected to reject the vast majority of negative examples. The classifiers become more complex as the cascading continues. Subsequent layers eliminate the additional negatives but require additional computation. After several stages of processing the number of sub-windows is reduced radically.

The detector is s 38 layer cascaded classifiers comprising 6060 features. The number of features in the first five layers of the detector is 1, 10, 25, and 50 respectively. The detection function returns a [N x 4] matrix, where N is the number of faces along with its position, width and height of the faces detected. If no face is detected it returns a variable with a value-1.

An outdoor image is shown in Fig. 5(a) and the detected faces using Viola Jones OpenCV Method are cropped as in Fig. 5(b).

2.2. Filtering Method

The detected faces from the face detector need to be filtered to remove the non facial images or false faces with the help of RGB and HSV colour spaces.

The filtering is done on the RGB and HSI values of the facial images. Of the two filters, RGB filter can represent the skin tones of dark complexion while the HSI filter can represent fair skin tones. Thus by combining the two filters a hybrid filter having higher efficiency than the individual ones is obtained. Figure 6 shows the block diagram of the filtering process employed.

(a) **(b)**

Figure 5. (a) Detected faces, and (b) Cropped faces.

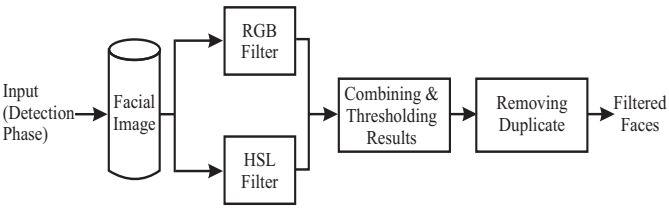


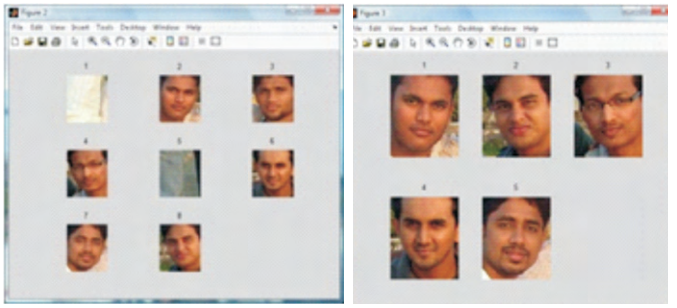
Figure 6. Filtering process.

3. FACE SEGMENTATION

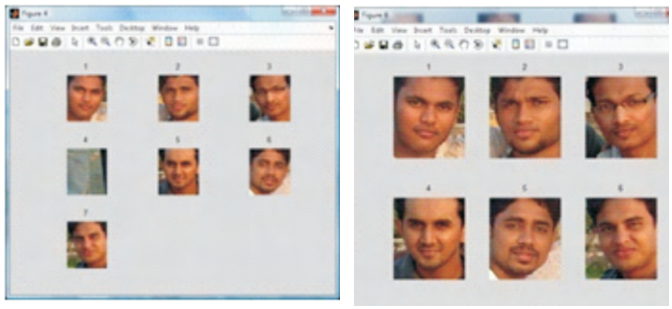
YC_bC_r colour space provides the chroma (C_rC_b) and luminance(Y) information in different channels[18]. In the case of the unknown lighting conditions, the light information might give rise to false detections, so the luminance component is discarded.

3.1 Skin Detection

The face detection is made by a decision function based on the Chroma component (e.g., C_rC_b from YC_bC_r or Hue from HSV). This requires the creation of the



(a) (b)



(a) (b)

Figure 7. (a) Detected faces, (b) Output of HSI filter, (c) Output of RGB filter, and (d) Output after combining and thresholding.

skin map by checking the conditions: (a) $140<C_r<165$, (b) $140<C_b<195$, and (c) $0.01<\text{Hue}<0.1$ for the existence of skin area. Skin areas are marked white while the non-skin areas are black as in Fig 8.

The detected face is cropped using the skin mask. Small background areas that could lead to errors will be deleted. A set of morphological operators is applied to remove the noise and to fill up the small holes. An open filter (erosion followed by dilation) removes small regions with 1's. These regions correspond to the noise and thin lines in the image. We then apply 2 dilation filters to connect the small regions together to form a face region. Erosion filters are used at the end to further eliminate the small islands and to disconnect any possible faces connected due to dilation.

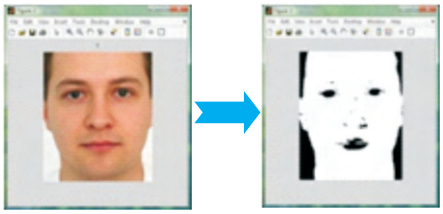


Figure 8. Extracted skin map.

3.2 Eye Detection

After the face skin map has been cropped, the next step is to locate the eyes from the face. A hybrid method[3] is favoured for the localisation of eyes in the facial images. This method utilises three cues, i.e., colour, edge and illumination to improve the accuracy. It is based on the observation that eye regions have dark colour, high density of edges and low illumination as compared to other parts of a face. The first step in this method is to extract the connected regions from the facial images using colour, edge density and illumination cues separately. Some of the regions are removed by applying the rules framed based on the shape of eyes. The remaining connected regions obtained through the three cues are combined to determine the candidate eye regions. The shape based rules are applied again to further remove the false eye regions.

3.2.1 Illumination-based Method

As part of the hybrid method, two separate eye maps, one from chroma component of the image and the other from the luminance component, are built up. The eye map from chroma with high C_b and low C_r values existing around the eyes is calculated from:

$$EyeMapC = \frac{1}{3}\left\{(C_b^2) + (\bar{C}_b^2) + \frac{C_b}{C_r}\right\} \quad (4)$$

where C_b, C_r and $\bar{C}_r$ are the normalised blue, red and negative of red chroma components respectively. To construct the eye map from luma, morphological operators (e.g. erosion and dilation) are designed to emphasise the brighter and dark pixels in the luma component in the eye regions. So the eye map from the luma component is calculated from:

$$EyeMapL = \frac{Y(x,y) \oplus g\sigma(x,y)}{Y(x,y) \ominus g\sigma(x,y)} \quad (5)$$

where $Y(x,y)$ is the luma component of the image and $g\sigma\ (x, y)$ is structuring element. The symbols $\oplus$ and Θ represent morphological dilation and erosion respectively. Figure 9 shows both chrominance and luminance maps of a face.

These two maps as shown in Fig. 9 are combined into a single eye map in Fig 10 using:

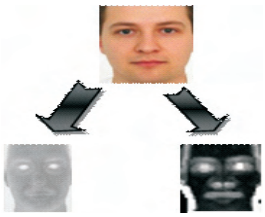


Figure 9. (a) Chrominance map and (b) Luminance map.

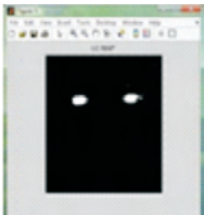


Figure 10. Eye map.

$$Image_{Illu}(EyeMapC)AND(EyeMapL) \quad (6)$$

3.2.2 Colour-based Method

This method is devised on the assumption that the eyes are the darkest regions of the face. Accordingly, the colour image is converted to a histogram equalised grey level image. The eyes regions are extracted from the equalised image using a threshold value of 20. Apart from the eyes, some other dark regions are also extracted as the connected regions. The unwanted regions can be removed via a component verification process. Figure 11 shows the EyeMap obtained from the colour-based method.

3.2.3 Edge Density Method

This method makes use of the fact that the eye regions possess more edge density than the other parts of the face. In this method, the facial image is converted to the grey level image and the edges are detected using Sobel edge detector. The dilation operator is applied twice to enhance the connected regions. The small holes inside the connected regions are filled up. Next the erosion operator is applied three times to get rid of the unwanted connected regions. A few shape and geometry based rules are invoked on the remaining connected regions to extract eyes as in Fig 12. This method works for all skin colours. However, it fails when the eyes are closed or the input image is poorly illuminated.

3.2.4 Hybrid Method

A hybrid method combines the above three methods to overcome the weaknesses of the individual methods. In this, blobs are detected using the three methods separately. Let , and be the binary blob images obtained through the illumination-based, colour-based and edge-density-based methods respectively, Any blob not satisfying the following rules is discarded:

(a) The solidity is of the region is greater than 0.5
(b) The aspect ratio is between 0.8 and 4.0
(c) The connected region is not touching the border
(d) The regions should be in the upper half of a face.

The bitwise AND operators are applied on all three blobs to obtain the connected regions as follows:

$$Image_{IlluCol} = (Image_{Illu})\text{AND}(Image_{Col}) \quad (7)$$

$$Image_{ColEdge} = (Image_{Col})\text{AND}(Image_{Edge}) \quad (8)$$

$$Image_{EdgeIllu} = (Image_{Edge})\text{AND}(Image_{Illu}) \quad (9)$$

The unwanted connected regions detected by one method but not by the other two are automatically removed due to the bitwise AND operation. The above three images are combined again using the bitwise OR operation as:

$$Image_{Hybrid} = (Image_{IlluCol})\text{OR}(Image_{ColEdge}) \text{ OR}(Image_{EdgeIllu}) \quad (10)$$

Shape and geometry based rules are applied on the hybrid image to get rid of the false eye. Figure 13 shows the schematic diagram of the hybrid method. Figure 14 shows the EyeMap obtained from the hybrid method.

3.3 Mouth Detection

To locate the mouth region, the presence of stronger red components and weaker blue components (C_r >

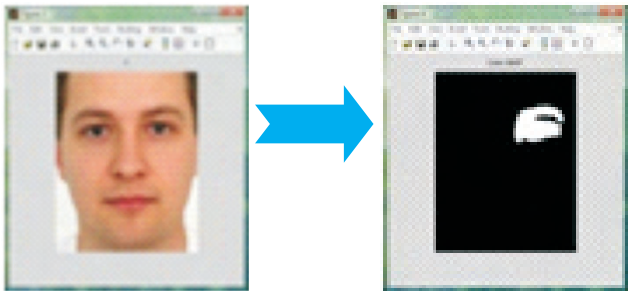


Figure 11. Colour EyeMap.

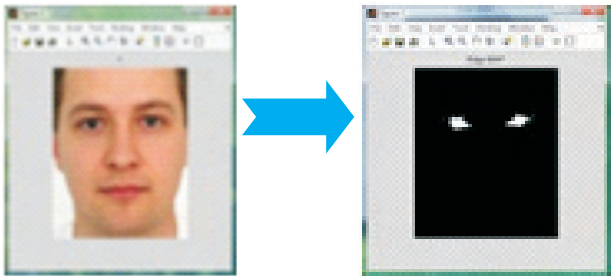


Figure 12. Edge density EyeMap.

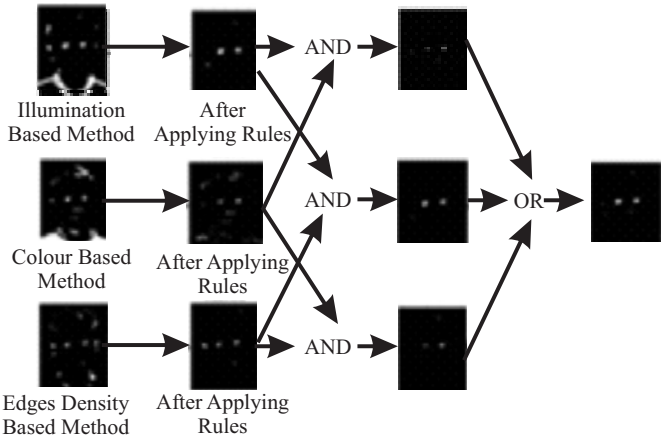


Figure 13. Hybrid method.

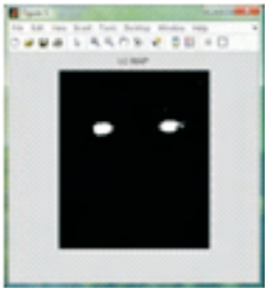


Figure 14. Hybrid EyeMap.

C_b) must be found to facilitate the construction of the mouth map[18]:

$$MouthMap = C_r^2 (C_r^2 - \frac{nC_r}{C_b}) \quad (11)$$

$$n = 0.95 \frac{\left(\frac{1}{k}\sum C_r(x,y)^2\right)}{\left(\frac{1}{k}\sum \frac{C_r(x,y)}{C_b(x,y)}\right)} \quad (12)$$

where k is the number of pixels on the face.

The surface plots (Figs. 15 and 16) are the result of application of Eqns. (11) and (12). The small areas around the large area in the surface plot in Fig. 15 constitutes the mouth region[18]. A threshold that is a little higher than the global minimum is chosen to locate the position of the mouth, which appears as in Fig 17. As can be seen there are still some errors due to more reddish skin parts (e.g. because of sunburn). This problem can be eliminated by applying different low-level image processing methods.

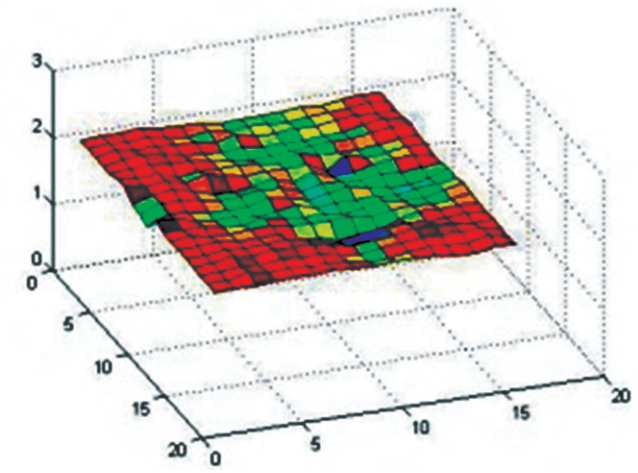


Figure 15. Surface plot of MouthMap.

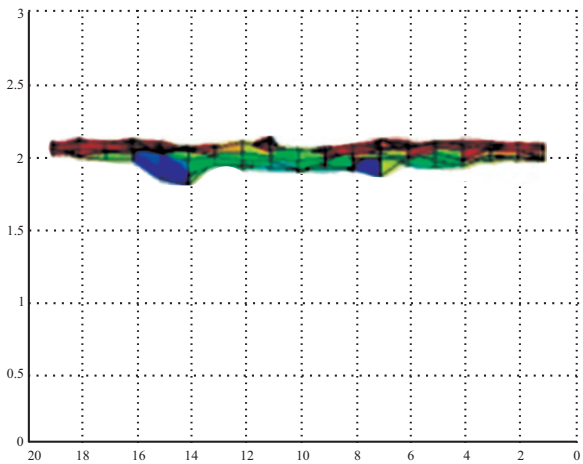


Figure 16. YZ Axis view of surface plot.

Knowing the location of the eyes, one can check whether they lie on a horizontal line; if not we must find the angle of its deferment (Fig.18). The mouth is then assumed to be aligned similarly. So we define a structuring element, i.e. a line, which is proportional to the face width, but oriented with an angle. Erosion deletes points that are smaller than the structuring element. A subsequent dilation with a line aligned the same way, but a little longer finally marks the mouth region.

Finally the extracted eyes and mouth regions from the given face are shown in Fig. 19.

3.4 Nose Detection

The eye section of the human face is segmented taking the upper and lower co-ordinates of the detected edges. In the same way the mouth is also segmented. After the segmentation of the eyes and mouth, the section between the lower co-ordinates of the eye and the upper co-ordinates of mouth is demarcated as the nose. This region in the gray level form is binarised. The output of the binarisation will be the white regions where the

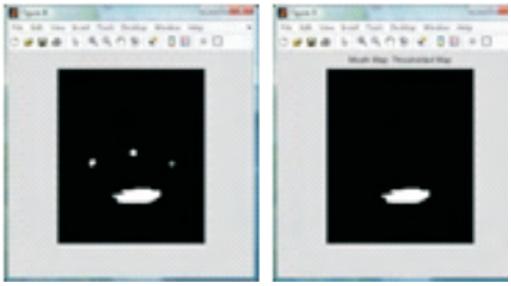


Figure 17. MouthMap.

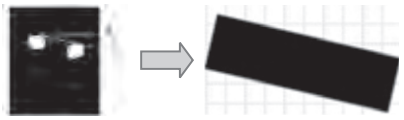


Figure 18. Structuring element.

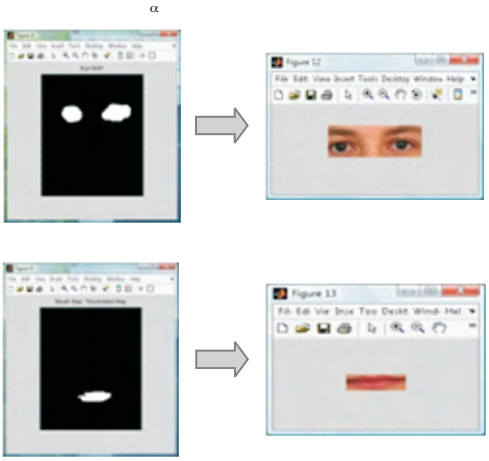


Figure 19. Eye and Mouth region extracted from (a) EyeMap and (b) MouthMap respectively.

intensity is higher than the threshold and the rest is black as illustrated in Fig. 20.

4. FACE ALIGNMENT

Once the eye map and mouth map are determined, we need to align the face if it is inclined, i.e. to make the nose parallel to the vertical axis and the eyes and mouth parallel to the horizontal axis. For this we compute the coordinates of the centre position between the two eyes in the eye map and the coordinates of the centre of the lower lip in the mouth map. The angle between the line joining these two points and the vertical axis passing through the middle of the face as in Fig. 21 is calculated as follows:

From Fig. 22, we have

sin α = opposite/hypotenuse = (b/a)

$$b = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_2)^2} \Rightarrow b = X_2 - X_1$$

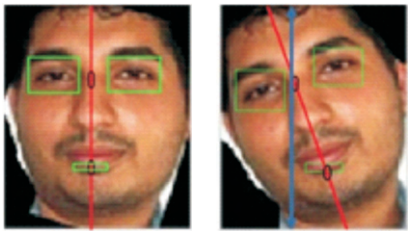
$$a = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \quad (13)$$

Using the angle α the face is rotated and aligned as shown in Fig. 23.

5. FORMULATION OF THE SEGMENTAL EUCLIDEAN DISTANCE CLASSIFIER

As part of the formulation, we now present extraction of PCA features from each segment of a face and the subsequent matching of features of the unknown (test feature vector) and the known (training feature vector) face images through the segmental Euclidean distances. In this study, we consider three faces along with their segments. Out of a database of faces some are taken as training set and the rest as the testing set. The differences between the training and testing feature vectors of each segment of the known and unknown faces are used to compute the segmental Euclidean distance that serves as the classifier. This approach is elaborated now.

Figure 20. Nose detection.



(a) (b)

Figure 21. (a)Aligned face, and (b) Inclined face.

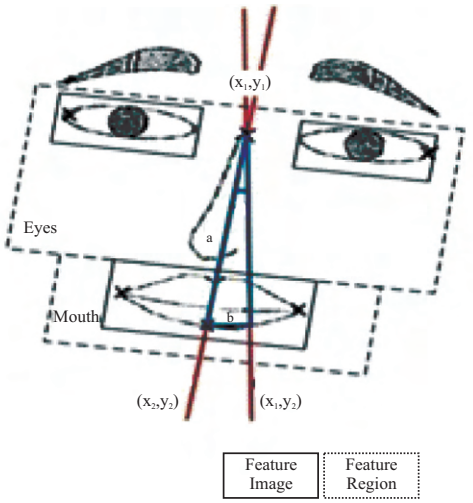


Figure 22. Face alignment.

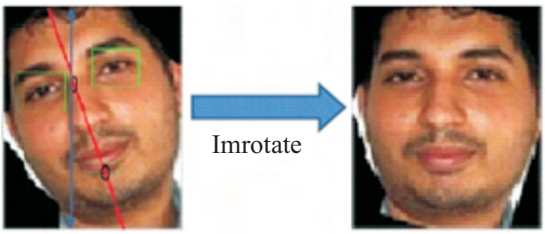


Figure 23. Corrected image

The segmented three faces of CG database of size 100 x 126 are shown in Fig. 24. Considering the five faces belonging to the training set we calculate the segmental Euclidean distances between the PCA features of the training set and those of a test (unknown) face.

To start with each data vector encompasses all the gray levels in a segment. Let a_i, b_i, c_i, d_i, e_i be the data vectors of sizes N x M_a = 100 x 32, N x M_b = 100 x 25, N x M_c = 100 x 25, N x M_d = 100 x 22 and N x M_e = 100 x 22 respectively of five segments of image1.Similarly we have for other 4 images. The extraction of PCA features from these data vectors is now discussed.

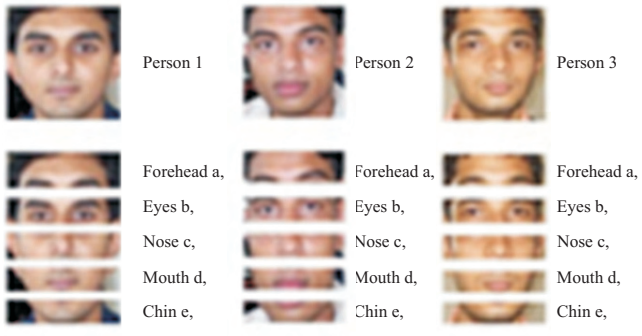


Figure 24. Segmentation of the faces for person *i*, *i* = 1.2.3.

5.1 Estimation of PCA features

For example a_i and b_i are two data vectors of two segments defined as:

$$a_i = \begin{bmatrix} a_{i1} \\ a_{i2} \\ \vdots \\ a_{i(N\times M_a)} \end{bmatrix} b_i = \begin{bmatrix} b_{i1} \\ b_{i2} \\ \vdots \\ b_{i(N\times M_b)} \end{bmatrix} i = 1,2,...,5 \quad (14)$$

The average data vector of the each segment of the face is computed as

$$m_{aik} = \frac{1}{5}\sum_{j=1}^{5} a_{jk} \quad m_{bik} = \frac{1}{5}\sum_{j=1}^{5} b_{jk} \quad m_{cik} = \frac{1}{5}\sum_{j=1}^{5} c_{jk}$$
$$m_{dik} = \frac{1}{5}\sum_{j=1}^{5} d_{jk} \quad m_{eik} = \frac{1}{5}\sum_{j=1}^{5} e_{jk} \quad (15)$$

where k = 1, 2,.....N x M_j

The average is subtracted from the data vector of each person to obtain

$$\overrightarrow{a_{mi}} = \begin{bmatrix} a_{i1} - m_{ai1} \\ a_{i2} - m_{ai2} \\ \vdots \\ a_{i(N\times M_a)} - m_{a_{i(N\times M_a)}} \end{bmatrix} \quad (16)$$

Next we build matrices: A_m, B_m, C_m, D_m, E_m of sizes (N x M_j) x P where j = *a, b, c, d, e* and P= number of persons in the training set.

For instance and have five vectors corresponding to the five users, given by

$$A_m = \left[\overrightarrow{a_{m1}} \ \overrightarrow{a_{m2}} \ \overrightarrow{a_{m3}} \ \overrightarrow{a_{m4}} \ \overrightarrow{a_{m5}}\right]; B_m = \left[\overrightarrow{b_{m1}} \ \overrightarrow{b_{m2}} \ \overrightarrow{b_{m3}} \ \overrightarrow{b_{m4}} \ \overrightarrow{b_{m5}}\right] \quad (17)$$

Similarly others matrices follow: $C_{m,}$ $D_{m,}$ E_m.

To reduce the dimension we construct another matrix V instead of covariance matrix for each segment such that their Eigen vectors are equivalent. These are computed as:

$$V_a = A_m^T.A_m; V_b = B_m^T.B_m; V_c = C_m^T.C_m;$$
$$V_d = D_m^T.D_m; V_e = E_m^T.E_m \quad (18)$$

We next evaluate the Eigenvalues and Eigenvectors S_j, from, V_j, j = *a,b,c,d,e*. The Eigen vectors of covariance matrix U_i are a linear combination of image space with the Eigen vectors. So we have

$$U_a = A_m.S_a; U_b = B_m.S_b; U_c = C_m.S_c;$$
$$U_d = D_m.S_d; U_e = E_m.S_e \quad (19)$$

The projection of each segment of face onto the face space yields the PCA feature vector:

$$\Omega_{ia} = U_a^T(\overrightarrow{a_{m1}}); \Omega_{ib} = U_b^T(\overrightarrow{b_{m1}}); \Omega_{ic} = U_c^T(\overrightarrow{c_{m1}});$$
$$\Omega_{id} = U_d^T(\overrightarrow{d_{m1}}); \Omega_{ie} = U_e^T(\overrightarrow{e_{m1}}); i = 1,2,..,N_{fj} \quad (20)$$

Using the above, PCA features of N_{fj}=5,5,5,5,5; for *j=a,b,c,d,e* are obtained for 5 segments of a face. For the purpose of matching the test Image is also segmented into 5 different segments: Forehead, eye, nose, mouth and chin. The corresponding five data vectors are:

$$\begin{bmatrix} r_{11} \\ r_{22} \\ \vdots \\ r_{(N\times M_j)} \end{bmatrix}; \text{j} = \text{a, b, c, d, e} \quad (21)$$

Next the differences between the data vectors and their average are calculated as

$$r_{mj} = \begin{bmatrix} r_{11} - m_{j1} \\ r_{22} - m_{j2} \\ \vdots \\ r_{(N\times M_j)} - m_{j(N\times M_j)} \end{bmatrix} \quad (22)$$

The projection onto the face space that corresponds to the PCA feature vector is computed from

$$\Omega_{rj} = U_j^T\left(\overrightarrow{r_{mj}}\right); \ j = a,b,c,d,e \quad r = 1,2,....N_{fj} \quad (23)$$

The distances between the features of the test face segments and the corresponding known face segments are:

$$\varepsilon_{ij} = \left\|\Omega_{ij} - \Omega_{rj}\right\|^2 \ j = a,b,c,d,e \ ; \ i=r=1,2,....N_{fj} \quad (24)$$

The aggregate segmental Euclidean distances of a training face Image with respect to a test face image are computed as

$$\varepsilon = \begin{bmatrix} e_a \\ e_b \\ e_c \\ e_d \\ e_e \end{bmatrix} = \begin{bmatrix} \Sigma\varepsilon_{ia} \\ \Sigma\varepsilon_{ib} \\ \Sigma\varepsilon_{ic} \\ \Sigma\varepsilon_{id} \\ \Sigma\varepsilon_{ie} \end{bmatrix} i = 1,2,..,N_{fi}$$

The aggregate Euclidean distance of a training wrt a test face is $\left[\Sigma_{p=a}^{p=e} e_p\right]^{\frac{1}{2}}$. The training face corresponding to the minimum of all aggregate Euclidean distances of all training faces with respect to a test face gives the identity of the test face. This procedure is repeated for all test faces to provide the overall success rate of the approach.

This segmental Euclidean method of face detection using PCA features (Eigen face) has been implemented on the CG database created from the databases of California Institute and Georgia Institute. This database consists of 413 images of 59 persons with 7 images for each person, out of which 5 for the training and 2 for the testing. The recognition rate with this approach is as 88 per cent, but it fares badly with a recognition rate of 70 per cent on the ORL database consisting of 40 persons with 10 images per person. In this we have taken 5 images for the training and 5 for the testing. For the sake of possible improvement in the results, we have tried other features such as fuzzy features and discrete cosine transform (DCT) features using a well known classifier called support vector machine (SVM) on both databases.

5.2 Estimation of Fuzzy Features

We are mainly interested in capturing the uncertainty in the sub images as faces contain smooth textures.

One way is to partition face image into windows of appropriate size and treat the gray levels in a window as a fuzzy set and fit a membership function. The gray levels along with their membership values are aggregated to yield a fuzzy feature. The formulae are given as part of the algorithm here. One could use different membership functions for the fuzzification of information contained in a window. We will use the membership function which is arbitrary. The Fuzzy features derived from the use of this membership function are now described.

An algorithm for the fuzzy feature extraction

1. The given face image is divided into a number of non- overlapping windows of size *W* x *W*.
2. The average intensity I_{avg} in k^{th} window is found using:

$$I_{avg}(k) = \frac{1}{W \times W} \sum_{i=1}^{W} \sum_{j=1}^{W} I(i, j) \quad (26)$$

3. The maximum intensity, Imax in the k^{th} window is found.
4. The membership function value, μ_{ij} for every pixel in k^{th} window is computed from:

$$\mu_{ij} = \left(\frac{\left| I(i, j) - I_{avg} \right|}{I_{\max}} \right) \quad (27)$$

Note that the membership function in (27) has an arbitrary shape but captures the random texture quite effectively. The membership values are in the lower range (0 to 0.2); so we subtract it from unity to make them close to 1 in the range (0.8 to 1). In view of this we consider $\bar{\mu}_{ij} = 1 - \mu_{ij}$ as the membership function.

5. The fuzzy feature from kth window is computed from:

$$FF = \frac{\sum \sum \bar{\mu}_{ij} I(i, j)}{\sum \sum \bar{\mu}_{ij}} \quad (28)$$

The above value though crisp is called a fuzzy feature for it is derived using fuzzy logic. The number of fuzzy features is equal to the number of windows fitted in a ROI. As far as the extraction of fuzzy features of face is concerned, we consider four window sizes for the experimentation. The number of fuzzy features from each face image of ORL database with 3 x 3, 5 x 5, 7 x 7 and 9 x 9 windows comes out to 1100, 396, 208 and 120 respectively. The corresponding number of fuzzy of features on each face of CG database for the same sizes of windows comes out to 498, 2040,980 and 594 respectively.

5.3 Results Using Support Vector Machine

Application of SVM on CG and ORL databases using PCA features results in the recognition rate of 18 per cent and 21 per cent respectively. Having realised that PCA features do not go well with SVM, we have gone in for well known DCT and fuzzy features derived from each face image of both CG and ORL databases. The DCT feature vector is of size 64 x 5 for CG database. As can be noted from Table 1 the number of training images per person is the same (i.e. five) in both databases but the number of testing images differs. In ORL database we have 5 test images instead of 2 in CGI database. We have experimented with different kernel functions to see the possible improvement in the performance of SVM. The results of application of SVM are given in Table 1. Linear kernel seems to give consistently high recognition rates on both databases (91 per cent and 89 per cent) with DCT features.

Table 1. Results of application of SVM.

Matching using different kernel function of SVM	Recognition Rate(%) with DCT features	
	CG Database	ORL Database
	5 train + 2 test images	5 train + 5 test images
Linear	91	89
Degree 2 polynomial	89	88
Degree 3 polynomial	90	89

The fuzzy features are also extracted using different window sizes from face images of two databases and are tested for the authentication using SVM with different kernel functions. The accuracy rate is found out for both the databases and it has been noticed that it gives better results with fuzzy features. The results are shown in detail in Table 2 and 3. The best accuracy of 96.6 per cent is obtained for the ORL database with fuzzy features on a window size of 7 x 7. There is an improvement of 2 per cent in the recognition rates as observed in the results of CGI database with fuzzy features on the window size of 3 x 3 over the results of DCT features. ROC which is a false acceptance rate (FAR) *vs* genuine acceptance rate (GAR) plot is constructed using features and Euclidean distance as classifier to demonstrate the overall performance of the authentication system. It may be noted that for a FAR of 0.1 per cent we get GAR of 86 per cent.

Table 2. Results of SVM with fuzzy features on CG database.

Matching using different kernel functions of SVM	Recognition rate (%) with fuzzy features on CG Database(5 training + 2 test)			
	3 x 3 window size	5 x 5 window size	7 x 7 window size	9 x 9 window size
Linear	92.37	90.67	89.83	85.59
Degree 2 polynomia l	87.28	86.44	83.05	84.74
Degree 3 polynomia l	86.44	83.05	82.20	83.05

Table 3. Results of SVM with fuzzy features on ORL

Matching using different kernel functions of SVM	Recognition rate (%) with fuzzy features on CG Database(7 training + 3 test)			
	3 x 3 window size	5 x 5 window size	7 x 7 window size	9 x 9 window size
Linear	93.33	94.16	96.66	94.16
Degree 2 polynomia l	94.16	92.5	95.83	94.16
Degree 3 polynomia l	95	93.33	94.16	94.16

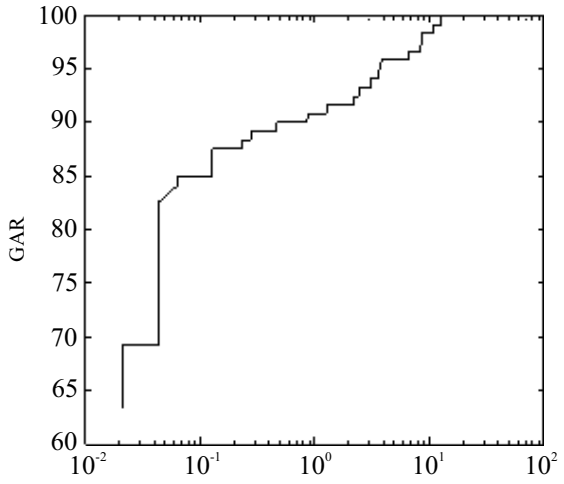


Figure 25. ROC using fuzzy features with a window size of 7 x 7 on ORL database.

6. CONCLUSIONS

A face recognition system is developed using 3 types of features (PCA, DCT and fuzzy) and two classifiers (Euclidean distance and SVM). This system is experimented on two databases using both segmental face and whole face. The debut of fuzzy features to the face recognition marks a significant contribution in this work.

A comprehensive methodology for the segmentation of a face into five segments, i.e. forehead, eyes, nose, mouth and chin has been developed. This methodology makes use of an Integral Image and cascade structured classifier which is built on Adaboost learning algorithm followed by the filtering process. The segmental Euclidean distance approach gives the recognition rate of 88 per cent with PCA features on CG database but does not perform well on the ORL database. Following this setback, we have explored both DCT and fuzzy features extracted from a whole face image using a library LibSVM version 3.0 of SVM classifier for the default values of parameters on two databases. The recognition rate is increased by 3 per cent with DCT features over PCA features on CG database. On the other hand, fuzzy features extracted with different window sizes on both the databases show further improvement in the recognition rates. ROC using fuzzy features on URL database with Euclidean distance as a classifier is plotted since SVM is not amenable to plot ROC as it can't give FAR and GAR but only the recognition rates. This performance as shown by ROC is somewhat inferior to that due to SVM.

It may be noted that the choice of appropriate features and classifiers is important to achieve good recognition rates on a particular database as borne out from this study.

REFERENCES

1. Turk, M. and Pentland, A. Eigenfaces for recognition. *J. Cognitive Neurosc.*, 1991, **3**, 71- 86.
2. Viola, Paul & Jones, Michael. Robust real-time face detection. *Int. J. Comp. Vis.*, 2004, **57**(2), 137-154.
3. Fleuret, Francois & Geman, Donald. Coarse-to-fine face detection. *Int. J. Comp. Vis.*, 2001, **41**(1/2), 85-107.
4. Ouna, E.; Freund, R. & Girosi, F. Training support vector machines: An application to face detection. 1997.
5. Papageorgiou, C.P.; Oren, M. & Poggio, T. A general framework for object detection. *In* Proceedings of International Conference on Computer Vision, 1998, 555-562.
6. Romdhani, S.; Torr, P.; Scholkopf, B. & Blake, A. Computationally efficient face detection. *In* Proceedings of International Conference on Computer Vision 2001, **2**, 695-700.
7. Rowley, Henry A.; Baluja, Shumeet & Kanade, Takeo. Neural network based face detection. *IEEE Tran. Pattern Analy. Machine Intel.*, 1998, **20**(1), 23-38.
8. Schneiderman, H. & T. Kanade, A statistical approach to 3D object detection applied to faces and cars. *In* IEEE Conference on Computer Vision and Pattern Recognition. 2000.
9. Kah Kay Sung & Tomaso Poggio, Example-based learning for view based human face detection. *IEEE Trans. Pattern Analy. Machine Intel.*, 1998, **20**(1), 39-51.
10. Nilsson, Mikael; Nordberg, Jörgen & Claesson, Ingvar. Face detection using local SMQT features and split up classifier. Blekinge Institute of Technology, 2005.
11. Stonham, T.J. Practical face recognition and verification with WISARD. Aspects of Face Processing, 1984, pp. 426-41.
12. Sung, K.K. & Poggio, T. Learning human face detection in cluttered scenes. *In* Computer Analysis of Image and Patterns. 1995. 432-39.
13. Lawrence, S.; Giles, C.L.; Tsoi, A.C. & Back, A.D. Face recognition: A convolutional neural-network

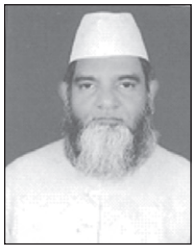
approach. *IEEE Trans. Neural Networks*, 1997, **8**, 98-113.

14. Weng, J.; Huang, J.S. & Ahuja, N. Learning recognition and segmentation of 3D objects from 2D images. *In* Proceedings of IEEE International Conference on Computer Vision. 1993, pp. 121-28.
15. Lades, M.; Vorbruggen, J.C.; Buhmann, J.; Lange, J.; Von DerMalsburg, C.; Wurtz, R.P. & Konen, M. Distortion Invariant object recognition in the dynamic link architecture. *IEEE Trans. Comp.*, 1993, **42**, 300-11.
16. Samaria, F. & Fallside, F. Face identification and feature extraction using hidden markov models. *In* Image Processing: Theory and Application, *edited by* G. Vernazza, Elsevier, 1993.
17. Tamura, S.; Kawa, H. & Mitsumoto, H. Male/Female identification from 8_6 very low resolution face images by neural network. *Pattern Recognition*, 1996, **29**, 331-35.
18. Maglogiannis, Ilias. Face detection and recognition of natural human emotion using Markov random fields. Personal and Ubiquitous Computing. 2007, Springer, London.
19. Shafi, Muhammad & Chung, Paul W.H. A hybrid method for eyes detection in facial images. *Inter. J. Electr., Compu. Syst. Engin.*, 2009, **3**(4).
20. Sirovich, L. & Kirby, M. Low-Dimensional procedure for the characterisation of human faces. *J. Optical Soc. Am.*, 1987, **4**, 519-24.
21. Kirby, M. & Sirovich, L. Application of the Karhunen-Loève procedure for the characterisation of human faces. *IEEE Trans. Analy. Machine Intel.*, 1990, **12**, 831-35.

Contributors

Mr Farrukh Sayeed has obtained his BTech and MTech from Aligarh Muslim University (AMU), Aligarh and National Institute of Technology (NIT) Jamshedpur, respectively. Presently pursuing PhD from Department of Electrical Engineering, Jamia Millia Islamia, New Delhi. He is a faculty in the Department of Electronics & Communication Engineering in P.A. College of Engineering, Mangalore.

Dr Madasu Hanmandlu
(Details available on page no. 430)



Dr Abdul Quaiyum Ansari received his BTech, MTech and PhD from AMU, Aligarh; IIT Delhi, New Delhi, and JMI, New Delhi, respectively. He is a Professor and Head, Department of Electrical Engineering, Jamia Millia Islamia, New Delhi. His research areas includes: Computer networks, image processing and fuzzy logic. He is on the advisory board and editorial board of many international and national Journals.