

A Face Detection Framework Based on Selected Face Components

Saeed Khoshfetrat Pakazad, Karim Faez and S. Masoud Nosrati

Abstract— In this paper a framework for fast face detection is presented. The features used in the system are low order Central Geometrical Moments (CGMs) of Face Components and their horizontal and vertical gradients. To speed up the detection process we have utilized a fast method to compute CGMs locally in the feature extraction phase, and in the classification phase we have used a fast multistage classifier. To enable each stage of the classifier to operate as fast as possible, in each stage, classification is carried out by using the optimal set of features which are selected for that particular stage according to a classification error measure. To detect faces in an image, a window the same size as the faces to be detected, scans the image and in each location the part of the image contained in the window is input to the multistage classifier which quickly discards background regions within its initial stages, and spends more computation on promising face-like regions. The presented results show that the proposed system yields good performance in terms of detection and false positive rates. The proposed framework is not limited to detecting faces and shall be used to detect other objects in an image as well.

I. INTRODUCTION

FACE detection is the first step to be carried out in applications involving human face processing and perception. Applications including face recognition/identification, face and/or facial feature tracking, facial expression recognition, which are utilized in human-machine interface systems, image data bases, teleconferencing, and security systems. A fast and accurate detection framework enhances the performance of a face processing system and the amount of post-processing required. A thorough survey of previous works on face detection can be found in [2] and [3]. Many of the recent methods focus on learning-based techniques that are data driven [1], [4], [14], [17]-[19]; in these methods classification function or face model is learned from a set of training images which capture the representative variability of facial appearance [3]. Our system shall be regarded as a learning-based system. In many systems classification is done based on pixel gray values, rather than features extracted from them [4], [14] [17], [18]. There are many motivations for using features rather than the pixel values directly; the most common reason is that features can act to code ad-hoc domain knowledge that is difficult to learn using a finite quantity of training data. Moreover, a feature-based system can operate much faster than a pixel-based system [1]. In our previous work [27] we

presented a face detection framework based on low order 2-dimensional Central Geometrical Moments (CGMs) of Face components and their horizontal and vertical gradients. Two-dimensional Moments [25] have been widely used in computer vision; typical examples of applications involving lower order moments in pattern recognition can be found in [20], [21]. The motivation behind using local object components is that they yield better results than Global features [15]. Since the detection process includes two major steps, i.e., feature extraction and classification, in order to speed up the detection process, in our previous system we proposed a fast method to extract local CGMs rapidly, and a fast multistage classifier [27]. Several fast algorithms have been described for computing geometrical moments for gray-level images [7], [22], [23], [24]; many of these methods compare their computational costs with Hatamian's digital filter method [7], and usually have computational costs comparable to Hatamian's method, specially for low order moments. In our previous paper [27] we propose a fast method to compute low order local geometrical moments, and we showed that its computational cost is much less than that of an improved version of Hatamian's method [8]. Moreover, unlike most of the other methods, its computational cost is invariant to scale (the size of the local window over which the moments are to be computed). In this system CGMs up to the order three are computed for object components, and to increase the accuracy of the system, we increase the number of components instead of using higher order moments. In our previous system in order to avoid round-off errors in the fast method proposed to compute local CGMs, the input image had to be resized to less than 320-by-240 pixels, but in the new system we have overcome this limitation by improving the proposed method by utilizing the Kahan summation algorithm [28], and the effect of round-off error [10] is decreased greatly.

To speed up the classification process, we use a multistage classifier in the system. Multistage or cascade classifiers have been utilized for fast face detection [1], [11]. A multistage classifier consists of a successively more complex classifiers combined in a cascade structure; many of the scanning windows are quickly rejected by early stages of the cascade and the few promising (face-like) ones can make their way to the subsequent stages which do more complex processing on them; therefore, a multistage classifier acts like an attentional filter (a focus of attention filter) [1] and speeds up the classification process. A great majority of the input image which contains non-face patterns is processed by the few initial stages of the cascade classifier, and only a few face-like parts of the image can pass the initial stage and reach the higher stages. Therefore, in order to make the

Saeed Khoshfetrat Pakazad and S. Masoud Nosrati are MSC students in the Pattern Recognition and Image Processing Lab., EE Dept., Amirkabir University of Technology, Tehran, Iran, 15914 (E-mails: saeed.kh.a@gmail.com, masoudnosrati@aut.ac.ir).

Karim Faez is with Electrical Engineering Department and Pattern Recognition and Image Processing Lab., Amirkabir University of Technology, Tehran, Iran, 15914. (E-mail: kfaez@aut.ac.ir).

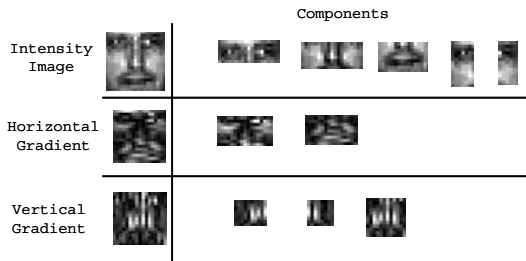


Fig. 1. Some possible Components of a face image and its horizontal and vertical gradients.

cascade classifier operate fast, the initial stages should be designed as simple as possible, i.e., each stage classifier should be designed so that it can reach the required false positive and true positive (detection) rates by using the minimal number of features. In our previous system [27] the feature for each stage of the classifier were chosen heuristically; in the system presented in this paper the best minimal set of features are selected for each stage by using a feature selection algorithm which selects the best features among the huge number of possible features, according to a classification error measure. By using these selected features for each stage, the average number of operations per location required for detection decreases greatly, compared to our previous system in which the features for each stage were selected heuristically.

The remaining sections of the paper are organized as follows. Section 2 gives an overview of the system. In section 3 the features and the proposed fast method to extract them locally and how to reduce round-off errors are discussed. In section 4 the proposed feature selection algorithm and the training procedure of the multistage classifier are explained. Section 5 is dedicated to results, and finally the paper ends with conclusions in section 6.

II. OVERVIEW OF THE SYSTEM

This system is intended for detecting multiple faces with different scales in a gray-scale image. Detection is carried out by shifting (sliding) a multi-scale window over the image and classifying the part of the image contained in the window as being either face or non-face.

Each window first goes through a gradient filter which rejects windows whose mean vertical or horizontal gradient is less than a certain threshold. Many parts of the image with nearly constant gray levels are quickly rejected by this filter and are not further processed. If a window is rejected by this filter, the system proceeds to the next location; otherwise the window goes through a multi-stage classifier which determines if it is showing a face or not. A multi-stage classifier consists of a series of classifiers, structured in a cascade; if a window is rejected (detected as non-face) by any stage, the system proceeds to the next window; otherwise the window will go to the next stage. The location of a window which manages to pass through all stages of the multi-stage classifier is marked by the system, indicating a face location.

At each stage of the classifier, CGMs for the selected components associated with that stage are computed.

CGMs are then normalized for intensity and scale, by dividing each of them by mean intensity of the window and S_w^{p+q+2} respectively [26], in which p and q are the order of the moment and S_w is the size of the window. Then the classifier of that stage (which in our system is an MLP Neural Network) classifies the window according to its computed features. The size of the window is gradually increased by a factor of 1.2, after each complete scanning over the image, i.e., $S_w = 1.2^s$ with s varying from -3 to 7 in our system; therefore with a base window size of 19-by-19 pixels, which is the size of the images in the training data base that we used, faces as small as 11-by-11 and as large as 68-by-68 pixels will be detected by the system. After completing the search over the image for all scales, usually multiple detections occur for each face at contiguous locations; these multiple detections are then merged to determine the exact locations of faces contained in the subject image.

III. FEATURES AND THEIR FAST COMPUTATION

The features used in the system are Central Geometrical Moments (CGMs) of Face Components and their horizontal and vertical gradients. Face Components are irregular rectangular regions on the face image and its horizontal and vertical gradients, as shown in fig.1. Detection is carried out by shifting (sliding) a window with the same size as the faces to be detected, over the image and classifying the part of the image contained in the window as being either face or non-face based on the value of the CGMs of the Components inside the scanning window. Therefore, at each location of the image, CGMs for several Components inside the scanning window should be computed. In our previous paper [27] we presented a fast method to compute CGMs for any rectangular region in the subject image, with the number of calculations required, being independent from the size of the region. And, it was shown that our method is much faster than Hatamian's digital filter approach for fast moment computation [7] and its improved version [8], for a PC-based system [9]. In our previous method the subject image had to be scaled down to less than 320-by-240 pixels (while preserving its aspect ratio, to avoid round-off errors that arose while computing CGMs. Round-off errors are caused because of the limited number of bits used for representing floating point numbers in computers. In this section after reviewing our previous method, we discuss the Kahan [28] summation algorithm and incorporate it into our previous method to decrease the effect of round-off errors.

A. Computing CGMs for Face Components using IMMS

Our face detector classifies the contents inside the scanning window as either face or non-face, based on the value of Central Geometrical Moments (CGMs) computed for the Components of that section of the subject image contained in the scanning window, and the Components of its horizontal and vertical gradients. Gradients are computed by using 3-by-3 vertical and horizontal Sobel

filters. In fig.1 a face image and some possible corresponding components are shown. Components are actually a number of sub-windows inside the window. For each stage of the multi-stage classifier different sets of components may be used, also the number of components usually increases in higher stages of the classifier because those stages have to solve a more difficult problem than that of previous stages; since each stage of the classifier is trained by using false positives of the previous stages; training procedure is described in the next section.

We have selected the best components for each stage of the classifier in this paper to optimize the speed of the classifier. There exist numerous methods for feature subset selection which shall be used for finding the best components for each stage of the classifier. Since the number of possible components among which the best components should be selected is very large, a fast subset selection method is presented in the next section. One may find a survey of feature selection methods and their applications in [5] and [6].

In the remaining of this subsection, we review our previously proposed method for fast local computation of CGMs and in the next subsection we discuss how to alter this method to reduce the round-off error.

Geometrical Moments, $m_{p,q}$, and Central Geometrical Moments of the order $(p+q)$, $\mu_{p,q}$, for a rectangular region R , in image I (see fig. 2) are defined respectively, as follows:

$$m_{p,q}(R) = \sum_{x,y \in R} x^p y^q I(x,y) \quad (1)$$

$$\mu_{p,q}(R) = \sum_{x,y \in R} (x - \bar{x})^p (y - \bar{y})^q I(x,y) \quad (2)$$

In (2), \bar{x} and \bar{y} refer to the coordinates of the center of the rectangular region.

Calculating CGMs up to the order three by using (1) requires $(10wh)$ additions and $(20wh)$ multiplications, in which w and h are width and height of the region, respectively, as shown in fig. 2; the total number of operations sums up to 12000 operations for a 20×20 pixel region and the number of operations increases quadratically with the size of the region. Such a large number of operations per location can make a multi-scale and exhaustive (pixel-by-pixel) search, infeasible for a fast detection system. Whereas the method proposed here, enables the system to perform an exhaustive multi-scale search with less than only 500 operations for calculation of CGMs up to the order three for any region in the image, independent of its size. Our method was in part inspired by Integral Image concept presented in [1] and the fact that Computation of Geometrical Moments for any region requires a double-summation over that region. The method begins with initial computation of 16 matrices (one for each order of the moment up to three), where each matrix is denoted by $IMM_{p,q}$ (Integral Moment Matrix of order $(p+q)$). These matrices are stored in memory, and by

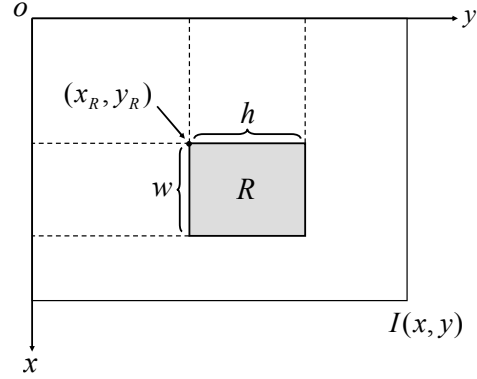


Fig. 2. Rectangular region R in image I , with its top left corner denoted by (x_R, y_R) .

using them, Geometrical Moments, $m_{p,q}$, can be computed for any region in the image, independent of its size in a few operations (only three additions/subtractions). After computing Geometrical Moments for the region of interest, Central Geometrical Moments are obtained by utilizing the binomial expansion of (2).

Each order of the matrix $IMM_{p,q}$ is a cumulative matrix having the same size as the input image I , defined as follows:

$$IMM_{p,q}(x,y) = \sum_{i=1}^x \sum_{j=1}^y i^p j^q I(i,j) \quad (3)$$

Each order of $IMM_{p,q}$ can be computed in one pass over the subject image by using an intermediate matrix $MM_{p,q}$ (Moment Matrix), and a pair of recurrences, formulated as follows:

$$MM_{p,q}(x,y) = x^p y^q I(x,y) \quad (4)$$

$$r(x,y) = r(x,y-1) + MM_{p,q}(x,y) \quad (5)$$

$$IMM_{p,q}(x,y) = IMM_{p,q}(x-1,y) + r(x,y)$$

In (5), $r(x,y)$ is the cumulative sum of rows of $MM_{p,q}$, and $r(x,-1) = 0$, $IMM_{p,q}(-1,y) = 0$.

After computing all $IMM_{p,q}$ matrices up to the order three, Geometrical moments, $m_{p,q}$, for any region R in the subject image (see fig. 2) can easily computed as follows:

$$\begin{aligned} m_{p,q}(R) = & IMM_{p,q}(x_R + w - 1, y_R + h - 1) \\ & + IMM_{p,q}(x_R - 1, y_R - 1) \\ & - IMM_{p,q}(x_R + w - 1, y_R - 1) \\ & - IMM_{p,q}(x_R - 1, y_R + h - 1) \end{aligned} \quad (6)$$

Then, Central Geometrical Moments of the region R can be computed by utilizing the binomial expansion of (2); as

illustrated in the following equation:

$$\begin{aligned}
 \mu_{pq}(R) &= \sum_{x,y \in R} (x-\bar{x})^p (y-\bar{y})^q I(x,y) \\
 &= \sum_{m=0}^p \sum_{n=0}^q [(-1)^{p-m} \binom{p}{m} \bar{x}^{p-m} (-1)^{q-n} \binom{q}{n} \bar{y}^{q-n}] \\
 &\quad \sum_{x,y \in R} x^m y^n I(x,y) \\
 &= \sum_{m=0}^p \sum_{n=0}^q [(-1)^{p-m} \binom{p}{m} \bar{x}^{p-m} (-1)^{q-n} \binom{q}{n} \bar{y}^{q-n}] m_{m,n}(R) \\
 &= \sum_{m=0}^p \sum_{n=0}^q CGMcoef(m,n,p,q) \bar{x}^{p-m} \bar{y}^{q-n} m_{m,n}(R)
 \end{aligned} \tag{7}$$

B. Decreasing round-off error by using Kahan summation algorithm

Basically, when a small number is added to a very large number, as what occurs in recursive cumulative sum of (5), the lower order bits of the small number are lost, due to limited number of bits dedicated for representing floating point numbers. The Kahan summation algorithm [28] works by introducing a correction factor to the summation. Suppose we are going to compute the summation:

$$S = \sum_{i=1}^n x_i \tag{8}$$

The Kahan algorithm to perform the above summation is shown fig.3. Each time a summand is added, there is a correction factor C which will be applied on the next loop. The lower order bits of y which are lost in the summation are recovered in C. So, in the next loop first, the correction factor is subtracted from the summand and then the summation is carried out. There is subtle point in the Kahan algorithm; that is, the correction factor, C, is not subtracted from T immediately, but it is subtracted from the summand in the next loop! This is because if we subtract the correction factor from T right away, again the round-off error occurs in the subtraction, because T is a large number and C is a small number.

It can be proved that the summand perturbation bound

<pre> // Computing r(x,y) for x = 1 to H { r(x,1) = MM_{p,q}(x,1) C = 0 for y = 2 to W { k = MM_{p,q}(x,y) - C T = r(x,y-1) + k C = [T - r(x,y-1)] - k r(x,y) = T } } </pre>	<pre> // Computing IMM_{p,q}(x,y) for y = 1 to W { IMM_{p,q}(1,y) = r(1,y) C = 0 for x = 2 to H { k = r(x,y) - C T = IMM_{p,q}(x-1,y) + k C = [T - IMM_{p,q}(x-1,y)] - k IMM_{p,q}(x,y) = T } } </pre>
(a)	(b)

Fig. 4. Computing the Integral Moments Matrices, by incorporating the Kahan summation algorithm to reduce the round-off error. a) first Computing the accumulative row sum of $MM_{p,q}$, b) and then computing the accumulative column sum of r .

$$\begin{aligned}
 S &= x(1) \\
 C &= 0 \\
 \text{for } j &= 2 \text{ to } N \{ \\
 &\quad y = x(j) - C \\
 &\quad T = S + y \\
 &\quad C = (T - S) - y \\
 &\quad S = T \\
 &\}
 \end{aligned}$$

Fig. 3. The Kahan Summation Algorithm.

decreases from $N\mathcal{E}$ to $2\mathcal{E}$ by using the Kahan summation algorithm, with N and \mathcal{E} being the number of summands in the summation, and the machine epsilon respectively [28].

By incorporating the Kahan algorithm into the recursive equation of (5), the round-off error shall be decreased. In (5) $r(x,y)$ is the cumulative sum of rows of $MM_{p,q}$ and $IMM_{p,q}$ is the cumulative sum of columns of $r(x,y)$; therefore, we can first compute the $r(x,y)$ matrix and then compute the $IMM_{p,q}$ matrix from it. The pseudo code of the altered recursive summation of (5) is shown in fig.4. In the pseudo code W and H are the width and height of the subject image respectively.

IV. MULTISTAGE CLASSIFIER AND FEATURE SELECTION

A multistage classifier is a cascade of classifiers at each stage of which a classifier is trained to detect almost all objects of interest while rejecting a certain fraction of the non-object patterns, and since the overall detection and false-positive rates of the multistage classifier is the product of those of individual stages, high detection rates and very low false-positive rates shall be obtained from cascade classifiers [1], [11]. Moreover, a multistage classifier attempts to reject as many non-face windows as possible, at the earliest possible stages, and since an overwhelming majority of scanned windows belong to

non-face class, this structure *increases the speed* of the system drastically [1], [11]. A great majority of the input image which contains non-face patterns is processed by the few initial stages of the cascade classifier, and only a few face-like parts of the image can pass the initial stage and reach the higher stages. Therefore, in order to make the cascade classifier operate fast, the initial stages should be designed as simple as possible, i.e., each stage classifier should be designed so that it can reach the required false positive and true positive rates by using the minimal number of features. In our previous system the feature for each stage of the classifier were chosen heuristically; whereas, in the system presented in this paper the best minimal set of features are selected for each stage by using a feature selection algorithm which selects the best features among the huge number of possible features, according to a classification error measure. By using these selected features for each stage, the average number of operations per location required for detection decreases greatly, compared to our previous system in which the features for each stage were selected heuristically. In subsection A we describe the feature selection procedure used to select the best components for each stage, and in the subsection B the training procedure of the cascade classifier is describes.

A. Feature Selection

For each stage of the classifier the best features are selected by using the false positives of previous stages, which constitute the non-face class, and the face samples of MIT CBCL data base [16]. The false positives are gathered by scanning several images containing no faces, like, natural scenes and different textures. The features used in the system are CGMs of Face Components and their horizontal and vertical gradients. The number of possible components for an image tends to be very large; therefore,

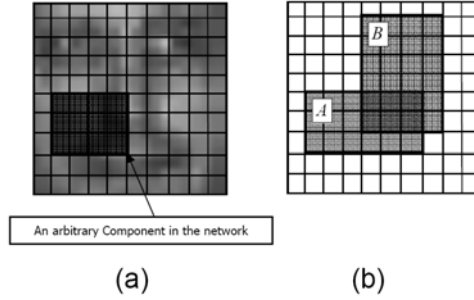


Fig. 5. a) Components in the overlaid network on the image. b) Two arbitrary overlapping Components.

in order to make feature selection feasible, a 10by10 network is overlaid on the images and all possible components in this network are considered for feature selection (see fig. 5a). The total number of components in the networks equals 3025 for each of which 16 CGMs should be computed; and this should be done for horizontal and vertical gradients as well. Therefore a total number of $3 \times 16 \times 3025 = 145200$ features should be evaluated to select the best features at each stage. Due to the large number of features to be evaluated, an aggressive feature selection algorithm is presented here which works based on a classification error criterion. The feature selection algorithm works as follows: first for each single feature, its value is computed for all the samples in the face and non-face classes, and then the histograms of the feature values for each class are computed. By using these two histograms, the classification error corresponding to the feature being evaluated is determined, by computing the Bayes error as the classification error measure. The features with smaller Bayes errors should be selected for classification, but it should be noted that features are correlated and selecting correlated features would not decrease the classification error since they are redundant [5], [6]; therefore after computing the Bayes error for all the features, they are clustered based on the spatial correlation of components and their Bayes error. Each cluster would contain features with similar Bayes error whose components have a high spatial correlation value. Then the required number of features for training each stage's neural network is picked out from the clusters with least Bayes errors. The spatial correlation of components is simply defined as their spatial overlap, which is equal to the ratio of cardinality of the intersection of two components' pixel sets to the cardinality of the union of their pixel sets; for the two components shown in fig. 5b, their spatial correlation shall be defined by:

$$C_{A,B} = \frac{|A \cap B|}{|A \cup B|} \quad (9)$$

The flowchart of the feature selection algorithm is shown in fig. 6. In the flowchart, \mathcal{E}_C , \mathcal{E}_S and $C_{C,S}$ denote the Bayes errors of and spatial correlation between the Current feature and the Seed of the current cluster respectively. Also, \mathcal{E}_{max} and C_{min} are constants defining the maximum allowed Bayes error difference and minimum allowed spatial correlation between cluster members; the values

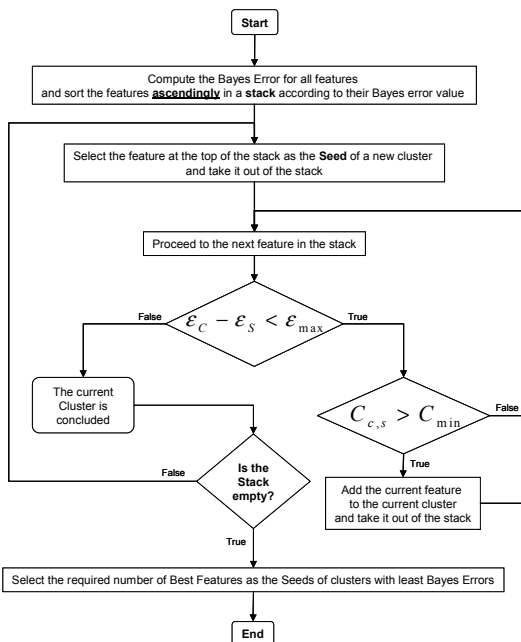


Fig. 6. The flowchart of the feature selection algorithm.











Component	Bayes Error	Moment Order (p,q)	# of Operations to compute the moment	Component	Bayes Error	Moment Order (p,q)	# of Operations to compute the moment
Brightness Features				Brightness Features			
	0.31448	(0,1)	14		0.2756	(1,2)	42
	0.31651	(0,1)	14		0.28774	(0,2)	21
Horizontal Gradient Features				Horizontal Gradient Features			
	0.3895	(1,1)	28		0.35926	(0,2)	21
	0.4044	(1,1)	28		0.38916	(1,2)	42
Vertical Gradient Features				Vertical Gradient Features			
	0.4503	(1,0)	14		0.31489	(0,0)	7

Fig. 7. Selected features for a) First stage, b) Second stage, classifiers, including the number of operations required to compute them.

chosen for these parameters in our experiments has been 0.1 and 0.5 respectively. In the results section the selected features for the first two stages of the classifier are presented.

B. Training Procedure of the Cascade Classifier

In a multi-stage classifier the overall detection (true positive) rate and false positive rates are the products of that of individual stages [1]. Therefore, by increasing the number of classifier stages, both the overall detection and false positive rates decrease; so, in order to have a high overall detection rate, each stage of the multi-stage classifier is trained so that it can achieve a detection rate near 100% while rejecting a certain fraction of non-face patterns. In a cascade structure each subsequent classifier is trained by using false positives of previous stages [1]. As a result, each subsequent stage faces a more difficult problem than that of the previous stages; therefore the number of features, which in our system depends on the number of Components, should normally be increased for each subsequent classifier to achieve a detection rate near 100%, while maintaining a reasonable rejection rate [1]. A detection rate near 100% shall be achieved by tuning the decision threshold of each stage classifier by observing its ROC (Receiver Operator Characteristics) curve. The multistage classifier in our system incorporates an MLP (Multi-Layer Perceptron Neural Network) with one hidden layer in each stage. The output of each stage's MLP is compared with a threshold, associated with that stage; input patterns which result in an output value greater than the threshold are classified as face, otherwise, as non-face. By varying the mentioned threshold from 1 to -1 and determining the detection and false positive rates, the ROC curve for the classifier in each stage is drawn. According to this curve the proper value for the threshold which results in a detection rate above 0.99%, is determined.

To train the MLPs we used the Resilient Back-Propagation algorithm [12] which provides much faster training than normal Gradient Descent algorithm [13]. Each element of the feature vectors (CGMs of Components) was normalized to zero mean and unity standard deviation prior to training; the target values for face and non-face classes were selected as 1 and -1 respectively. To train the first stage we used the CBCL MIT data base [16] and for subsequent stage, 2800 faces in CBCL MIT data base and the same number of false positives from previous stages were used. To gather the false positives the system was fed with several images containing no faces, like natural scenes or different textures, etc, and all the detections were considered as false positives.

One problem that may occur in the training of a Neural Network is over-fitting [13], i.e., the error on the training set is driven to a very small value by the training algorithm, but the network's classification error when presented with new data tends to be large and therefore the network demonstrates poor generalization. In order to avoid over-fitting we divided the available data set into two subsets, one for training and one for validation. The training was continued until the error on the validation set stopped decreasing. For each stage of the classifier prior to training the best features for that stage are selected by using algorithm presented in the previous subsection. The number of neurons and components normally increases for higher stages as described before (see table I).

V. RESULTS

In our experiments we trained a seven-stage classifier using the method described in the previous section. Table I, summarizes the parameters of each stage of the classifier after training. It can be seen that the complexity of each subsequent stage classifiers gradually increases in terms of the number of neurons and components, since higher stages

TABLE I
PARAMETERS OF EACH STAGE CLASSIFIER AFTER TRAINING

Stage #	# of Hidden Layer Neurons	Decision Threshold	Detection Rate (%)	False Positive Rate (%)	# of Components	# of operation to compute the components
1	11	-0.96	99.01	49.03	5	98
2	11	-0.92	99.23	31.20	5	133
3	14	-0.94	99.00	35.01	7	189
4	15	-0.94	99.18	28.11	10	210
5	17	-0.96	99.13	12.67	28	315
6	21	-0.92	99.25	8.09	35	420
7	23	-0.94	99.01	7.51	41	511

face a more difficult problem than early stages. Since the overall detection and classification rates of a multi-stage classifier is the product of that of individual stages [1], one can expect a detection rate of 93.97% with false positive rate of 1.15e-5 from the system, according to Table I. Nevertheless the false positive rate of the system can be decreased by adding more stages to the classifier. The selected features for the first two stages of the classifier are shown in fig. 7. It can be seen that different features are suitable for different stages of the multi-stage classifier. The average number of operations required to compute the features for classification at each location in the subject image can be obtained as follows [1]:

$$N = n_1 + \sum_{i=2}^K (n_i \prod_{j<i} f_j) \quad (10)$$

In the above equation n_i is the number of operations required to compute all the features in the i-th stage of the classifier, and f_i is the false positive rate of the i-th stage. The average number of operations to compute features per location according to table I equals 209 for the system; whereas, in our previous system [27] this quantity was about 2240 operations per location. Effectively, in the new system by using the best selected features for each stage, the average number of operations per location decreases considerably compared to our previous system in which the features for each stage were selected heuristically, therefore a significant speed increase is achieved in the new detection system.

VI. CONCLUSION

In this paper, we presented a fast face detection system which excels previous face detection [27] system in two major aspects. Firstly, we improved our proposed method for fast calculation of local CGMs, by incorporating Kahan summation algorithm [28] to decrease the effects of round-off error. Secondly, in this paper we proposed an aggressive feature selection algorithm to select best features for each stage of the multistage classifier; which enabled the system to achieve the desired detection performance by using fewer features, which in turn resulted

in a great reduction in the average number of calculations required to compute features per location, and therefore a significant speed advantage over our previous system is obtained.

Since the number of features among which the best ones had to be selected for each stage was very large, we proposed an aggressive feature selection algorithm based on Bayes error as the classification error measure and clustering to eliminate correlated features. There exist other feature selection methods whose performance shall be investigated for this problem and compared with our proposed method. Also, using other statistical classifiers for the stage classifier like SVMs (Support Vector Machines), and more efficient algorithms for training the cascade of classifiers could be examined in future works. The framework presented in this paper is general in that no prior assumptions are made on the type of the object to be detected by the system. Moreover, the feature set used in the system is a comprehensive one in that it includes both the intensity and gradient information of the object; this comprehensive feature set combined with the presented fast feature selection algorithm makes the proposed framework suitable for detecting other object types as well, as long as proper training data bases are available for them.

REFERENCES

- [1] P. Viola and M. -J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137-154, 2004
- [2] M.-H. Yang, D. J. Kreigman, and N. Ahuja, "Detecting Faces in Images: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 1, pp.34-58, 2002.
- [3] M.-H. Yang, "Recent Advances in Face Detection," *IEEE ICPR Tutorial, Cambridge, United Kingdom*, August, 2004. Available: <http://vision.ai.uiuc.edu/mhyang/face-detection-survey.html>
- [4] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Patt. Anal. Mach. Intell.*, vol. 20, pp.22-38, 1998.
- [5] M. Dash and H. Liu, "Feature Selection for Classification," *International Journal of Intelligent Data Analysis*, vol. 1, pp. 131-156, 1997.
- [6] E. Cantu-Paz, S. Newsam, and C. Kamath, "Feature Selection in Scientific Applications," in *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, pp. 788-793, 2004.
- [7] M. Hatamian, "A real-time two-dimensional moment generating algorithm and its single chip implementation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 3, pp. 546-533, 1986.

Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2007)

- [8] W.-H. Wong and W.-C. Siu, "Improved digital filter structure for fast moments computation," in *IEE Proc. Vis. Image Signal Process.*, vol. 146, No. 2, pp. 73-79, April 1999.
- [9] D. Alpert and D. Avnon, "Architecture of the Pentium Microprocessor," *IEEE Micro*, vol. 13, no. 3, pp. 11-21, May/June, 1993.
- [10] J. Martinez and F. Thomas, "Efficient Computation of Local Geometrical Moments," *IEEE Trans. On Image Processing*, vol. 11, no. 9, pp. 1102-1111, September 2002.
- [11] R. Lienhart, A. Kuranov, and V. Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection," *Lecture Notes in Computer Science*, vol. 2781, pp. 297-304, 2003.
- [12] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. of the IEEE International Conference on Neural Networks*, pp. 586-591, 1993.
- [13] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, Boston, MA: PWS Publishing, 1996.
- [14] K. Sung and T. Poggio, "Example-based learning for viewbased face detection," *IEEE Patt. Anal. Mach. Intell.*, vol. 20, pp. 39-51, 1998.
- [15] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio, "Categorization by Learning and Combining Object Parts," *Advances in Neural Information Processing System*, vol. 2, no. 14, pp. 1239-1246, 2001.
- [16] CBCL Face Data Base #1
MIT Center for Biological and Computation Learning
Available: <http://www.ai.mit.edu/projects/cbcl>
- [17] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proc. Of the IEEE conf. on Computer Vision and Pattern Recognition*, pp. 130-136, 1997.
- [18] H. Schneiderman and T. Kanade, "A statistical method for 3D object detection applied to faces and cars," *Proceedings of Int. Conference on Computer Vision and Pattern Recog.*, vol. 1, pp. 746-751, 2000.
- [19] D. Roth, M. Yang, and N. Ahuja, "A snowbased face detector," in *proc. Advances in Neural Information Processing Systems*, pp. 855-861, 2000.
- [20] M.-K. Hu, "Visual Pattern recognition by moment invariants," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 179-187, 1962.
- [21] Y. S. Abu-Mostafa et al, "Recognition aspects of moment invariants," *IEEE trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 698-706, 1984.
- [22] B. Li, "High-order moment computation of grey-level images," *IEEE trans. Image Processing*, vol. 4, pp. 723-730, 1992.
- [23] J. Martinez and F. Thomas, "A reformulation of gray-level image geometrical moment computation for real-time applications," in *IEEE conf. Robotics and Automation*, vol. 3, pp. 2315-2320, 1996.
- [24] T-W. Shen, D. P. K. Lun, and W. C. Siu, "Fast Algorithm for 2-D image moments via Radon transform," in *int. Conf. Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1327-1330, 1996.
- [25] C.-H. Teh and R. T. Chin, "On Image Analysis by the Methods of Moments," *IEEE Trans. On Pattern Recognition and Machine Intelligence*, vol. 10, no. 4, pp. 496-513, 1998.
- [26] A. Khotanad and Y. H. Hong, "Invariant Image Recognition by Zernike Moments," *IEEE Trans. On Pattern Analysis and Machine Vision*, vol. 12, no. 5, pp. 489-497, May 1990.
- [27] S. Kh. Pakazad, K. Faez and F. Hajati, "Face Detection Based on Central Geometrical Moments of Face Components," in *Proc. of the IEEE International Conference on Systems, Man and Cybernetics 2006*, to be published.
- [28] W. Kahan, "A Survey of Error Analysis," in *Information Processing 71*, vol. 2, pp. 1214-1239, 1972.