Real Time Face Object Detection for Security and Surveillance Application

Ramya N.
Jerusalem college of Engineering/Department of ECE, Pallikaranai,Chennai,India
Email: ramyanatarajan811@gmail.com

Durga.G
SSN College of Engineering /Department of ECE, Kalavakkam,Chennai,India
Email: durgag@ssn.edu.in

***Abstract*—Object detection deals with detecting the presence of objects such as human face, car, buildings, road symbol, etc.,. For surveillance and security based applications, automatic object detection is efficient than manual operation. Existing object detection techniques are limited to small sized images, and detect the human face at lesser frames per second. This paper focus on face objects detection of independent image size using Adaptive Boost algorithm. Adaptive Boost algorithm is a machine learning algorithm. It consists of a training phase, where the required threshold values are estimated from training images. In the training phase the rectangular sum of features is calculated to obtain an integral image. The use of the integral image is to compute the sum of the rectangular features rapidly. This is followed by a testing phase, where the estimated threshold values are used in the detection of face objects. It passes to many stages to detect the object. If the object in the image is larger than search window size and feature size, it will not detect the object. So the image is scaled and then the object is found. The algorithm has been extended to detect faces in videos as well. The algorithm is simulated in MatLab 7 to detect objects in any image with a frame rate that can vary for various applications and input image frame sizes.**

***Index Terms*—object detection, adaptive boost algorithm, face detection, haar-like feature**

I. INTRODUCTION

Object detection can be obtained by extracting the information from image, process the information and determining whether the object is found in exact location or not. It is used in several real time embedded application such as computer vision and image processing applications, bioinformatics, security, and artificial intelligence. Surveillance is the monitoring of physical behavior and activities of the people. The important application is human detection in CCTV surveillance system to prevent unauthorized access and identifying correct person in secure system, such as office, bank etc. There are several algorithm used to perform face object detection such as Supported Vector Machine (SVM), Adaptive Boost [8]. SVM is used to detect the frontal view human face. It fail to detect the rotational plane face in the images and accuracy is less when compared to other methods.

The proposed work focus on detecting the face objects in surveillance and security application using Adaptive boost algorithm. Adaptive boost algorithm is combined with any method for finding weak classifier as it quickly eliminate the non face region and the classification process itself is extremely fast([1], [2]). But it is data dependent and susceptible to noise. The real-time face detection application system is the ability to detect the face rapidly and increase the speed which is infeasible before [6]. The problem in this is, face detector is trained on frontal, upright faces and they are roughly aligned. So there will be a variation in rotation both in plane and out of plane. The rest of the paper is organized as follows. In Section II, detailed description of Adaptive Boost algorithm is given, Section III contains simulation result and the conclusion is given in Section IV.

II. ADAPTIVE BOOST ALGORITHM

A. Algorithm

Adaptive Boost algorithm is a machine learning method. It utilizes a small number of weak classifier. The weak classifiers are necessary to detect the object from the input image. The weak classifier is a classifier which only slightly correlated with true classification. Classification function can be determined by the algorithm which uses large amount of positive and negative images to train the weak classifier. It is used to boost the classification performance. Strong classifier can be constructed by combining many weak classifiers to result in high accuracy rate and computational efficiency. Linear combinations of weighted results of this weak classifier are represented as strong classifiers, and the weights for weak classifiers are trained with many positive and negative example images.

B. Block Diagram of Adaptive Boost Algorithm

Adaptive boost algorithm consists of five stages. They are image scaling, computation of integral image feature

computation, stage computation, and identifying the face objects shown in Fig.1. The Haar-like feature is used to calculate the weak classifier. The Adaptive Boost algorithm used features starting at 24 x 24 pixels or 19 x 19 pixels. The feature size can vary with the application and the number of rectangles for each feature variation also depends on object. The strong classifiers constructed by Adaptive Boost algorithm are setup in a cascade and each strong classifier is a stage in the detection process.

Each stage consists of a group of Haar-like features selected during training. Each Haar-like feature in a stage output is computed and accumulated. The input image is used to calculate the integral image and given to the feature and stage computation. The sum of all weighted feature rectangles is the feature sum. All feature sums is calculated in the search window. If this rectangle sum exceeds the threshold feature sum is set to a predetermined value obtained from the training set else feature sum is set to another predetermined value from the training set. All feature sums are accumulated to compute the stage sum. For each stage the corresponding stage threshold value is determined.

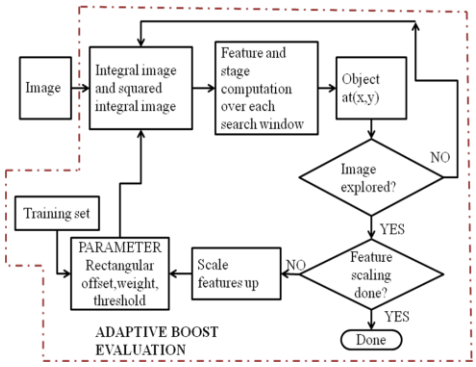

Figure 1. Block diagram of adaptive boost algorithm.

Stage threshold value is compared with stage sum, if it is greater than stage threshold corresponding search window is moved to the next stage, else the corresponding search window will be rejected. Because of this technique search window which does not contain face objects is removed quickly. If the face objects in the image frame are larger than the search window and feature size it will not detect the object. So image is scaled and detects the face object in the given image. The computation is repeated for both positive and negative training sets images shown in Fig. 2.

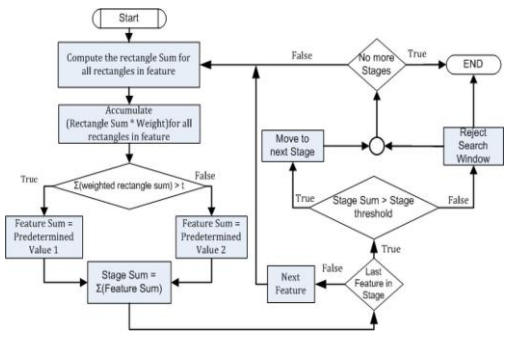

Figure 2. Adaptive Boost Based Classification [3]

C. *Haar-like Feature*

Haar-like feature is used to calculate simple features. For weak classifiers in the Adaptive Boost algorithm, this can detect an edge or a line feature and resemble the Haar basis function [5]. The Haar-like feature is a fixed image which contains small number of black and white rectangles This feature act as filter that can detect the presence and absence of certain characteristics in an image [3]. To calculate the Haar-like feature from an image is as following steps.

1. Sum of pixels values within black and white rectangle region using formula given below
   Haar-like feature = (sum of pixel in white region)-(sum of pixel in black region)
2. Difference of the weighted sum of these regions is calculated.
3. If the difference exceed predefined threshold value, then the weak classifier output become true and edge feature exists
4. If not weak classifier output is false
5. Combining this weak classifier, a boosted strong classifier that detect more complicated object is achieved. Examples for Haar-like features as shown in Fig. 3.

- 2-rectangle feature
- 3-rectangle feature
- 4-rectangle feature

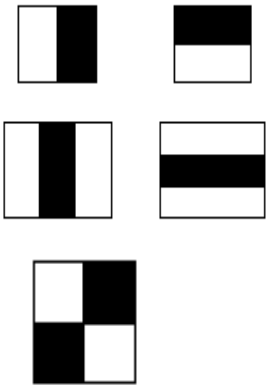

Figure 3. Haar –Like Rectangle Feature

The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and they are horizontally or vertically adjacent. A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a enter rectangle. A four-rectangle feature computes the difference between diagonal pairs of rectangles [6].

D. *Integral Image*

Rectangle feature are calculated from the integral image rapidly. The sum of the pixels above and to the left of point x, y is the integral image at the point (x, y).

$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x, y) \quad (1)$$

where $ii(x, y)$ is the integral image and $i(x, y)$ is the original image as shown in Fig. 3.2. Using the following pair of recurrences,

$$s(x, y) = s(x, y-1) + i(x, y) \quad (2)$$

$$ii(x, y) = ii(x-1, y) + s(x, y) \quad (3)$$

where $s(x, y)$ is the cumulative row sum, $s(x,1) = 0$, and $ii(-1, y) = 0$, the integral image can be computed in one pass over the original image. Using the integral image any rectangular sum can be computed in four array references shown in Fig. 4.

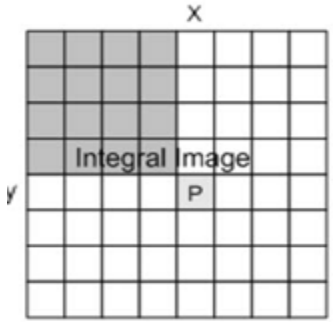

Figure 4. Integral Image [3]

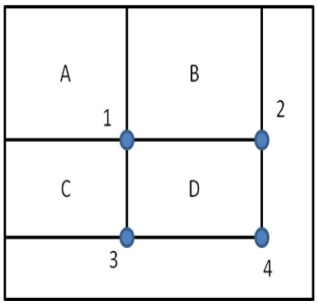

Figure 5. Rectangular Sum [3]

The Fig. 5 represents the sum of the pixels within rectangle D computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A+B, at location 3 is A+C, and at location 4 is A+B+C+D. The sum within D can be computed as(4+1)-(2+3). From this the difference between two rectangle sums are calculated using six reference points. Similarly for three rectangle and four rectangles are computed using eight and nine reference points respectively.

E. Training Sets

The training sets consist of face and non face image. All image size is equal such as 19 x 19 pixel, 24 x 24 pixel value Examples of face and non face images [4] are shown in Fig. 6 and Fig. 7

Figure 6. Face Image

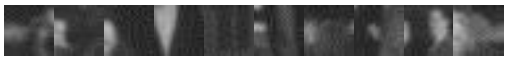

Figure 7. Non Face Image

A face object is a rectangular box of the original image called a sub-window. Generally these sub-windows have a fixed size (typically 19 x 19 pixels). The algorithm scans the entire image with this window and detects the each face object. A sub-window with size of 19 x 19, 24 x 24 pixels is used to move pixels for each step over a given image. For each positive and negative training sets image the Haar-like features are calculated which are shown in Fig. 8

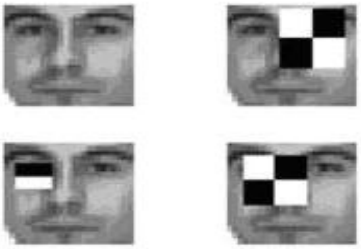

Figure 8. Examples of Calculating Haar-Like Feature for Training Image [7]

F. Feature and Stage Computation

A Haar feature classifier uses the rectangle integral to calculate the value of a feature as shown in Fig. 9. The classifier multiplies the weight of each rectangle box by its area and all the outputs are added together. Several classifier forms a stage. A stage comparator sums all the classifier results in a stage and compares this summation with a stage threshold. The threshold is a constant obtained from the Adaptive Boost algorithm. Each stage does not have a set number of Haar features. Depending on the parameters of the training data individual stages can have a varying number of Haar features.

Figure 9. Feature Computation [3]

Single feature vector f_i evaluated at x the output of the weak classifier $h_i(x)$ is either 0 or 1. The output depends on whether the feature value is less than a given threshold θ_i

$$h_i(x) = \begin{cases} 1, & if \sum_{t=1}^{T} f(x) < \theta_i \\ 0, & x \geq 0 \end{cases} \quad (4)$$

and x is the rectangle box to be classified. The set of weak classifier has the set of feature. From the evaluation of each feature type on training data, it is possible to estimate the threshold value of each classifier.

The algorithm steps is as follows
Given,

- Training set $(x_1, y_1), (x_2, y_2), \ldots\ldots\ldots (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{+1, -1\}$
- Number of iteration T,
- Initialize

$$w_1(i) = \frac{1}{m}, \frac{1}{n} \quad i = 1,2,3\ldots\ldots m \text{ or } n$$

where m is the number of positive image size and n is the number of negative image size

for $t = 1,2 \ldots T$

- Normalize the weight

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \tag{5}$$

- Find the classifier $h_i(x)$ from the family of weak classifier *H* that minimize the error with respect to w_i

$$h_t = \arg\min_{h_t \in H} \varepsilon_t \tag{6}$$

$$\varepsilon_t = \min_{f,\theta} \sum_i \left| w_i h(x_i, f, \theta) - y_i \right| \tag{7}$$

where f is feature of the rectangle sum value

- If $\varepsilon_t \geq 0.5$ then stop the process
- Choose $\alpha_t \in R$ typically $\frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t}$ where ε_t is the weighted error rate of classifier
- X_i classified incorrectly, don't change the weight, otherwise change the weight

$$w_{t+1} = w_{t,i} \left(\frac{\varepsilon_t}{1-\varepsilon_t} \right) \tag{8}$$

- Linear combination of weak classifier is strong classifier
- Output of final classifier

$$h(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) h_i(x) \geq \left(\frac{1}{2}\right) \sum_{t=1}^{T} \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

Thus, after selecting the final classifier h_t for the weights w_t the examples which are correctly identified are weighted less and those incorrectly identified are weighted more. Therefore the algorithm is used to test the classifier on the weights and it will select a classifier that better identifies those examples that the previous classifier missed [6].

G. Cascade Stages

Using the Adaptive Boost algorithm, constructing the cascade of classifier will obtain the strong classifier, which will quickly reject many of the sub-windows which do not contain the face object. The initial $\frac{1}{2} \sum_{t=1}^{T} \alpha_t$ Adaptive Boost threshold is designed to yield a low error rate on the training data. A lower threshold yields higher detection rates and higher false positive rates. Using the training sets, the classifier can be adjusted to detect 100% of the face objects and 50% of false positive rate [6].

The overall form of the detection process is a decision tree called as "cascade" structure shown in Fig. 10. The sub-window images are given to the cascade structure. The positive output of a first stage triggers the evaluation of a second stage used to achieve very high detection rate. A positive output from the second stage triggers a third classifier, and so on. A negative output outcome at any stages will be the immediate rejection of the sub-window. At the earlier stages the negative output are rejected as possible while the positive output will trigger the evaluation of every stage in the cascade structure.

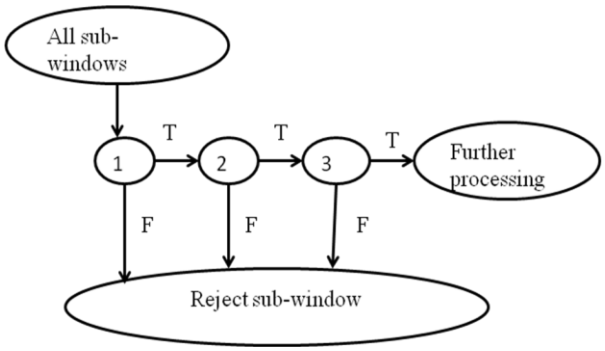

Figure 10. Cascade Classifier

III. SIMULATION RESULT

The simulation result is obtained from MatLab. Training positive and negative images are given. Here 10 positive images and 20 negative images are given for training set. Each training set calculate the Haar-like feature and obtained the strong classifier. Cascade classifier is used to remove the non face region while passing through every stage. Here we use 25 stage of cascade classifier. Each stage will calculate the feature. For example 2 stage cascades will calculate 2 features, 5 stages will calculate 20 features, etc…and the face is detected.

For the given various size input images, the face is detected using the Adaptive Boost algorithm as shown below.

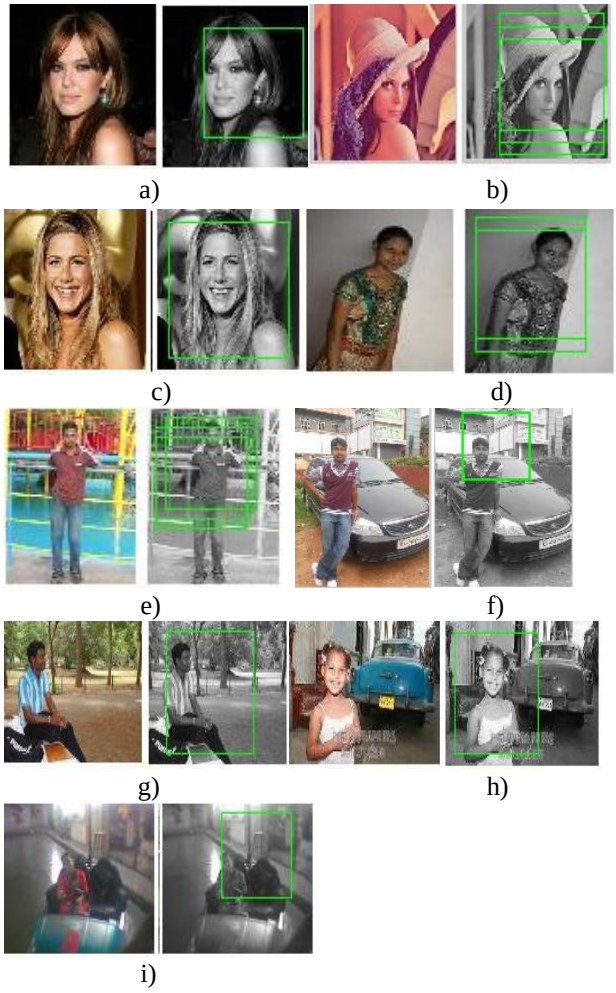

Figure 11. Input and Output image for a) 225 X 225 pixel image;b) 204 X 204 pixel image; c) 180 X 180 pixel image; d) 206 X 203 pixel image; e) 293 X 220 pixel image; f) 225 X 300pixel image; g) 240 X 180 pixel image; h) 183 X 275pixel image; i) 264 X 324 pixel image;

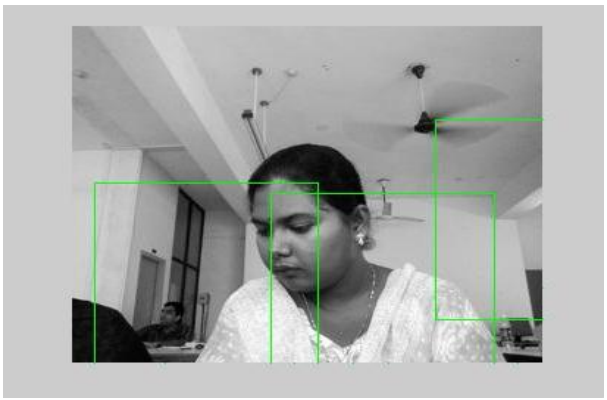

Figure 12. Video – Single Face Image

For the video input image, face is detected for less than or equal 30 frames per second as shown in Fig. 19 and Fig. 20 using the Adaptive Boost algorithm

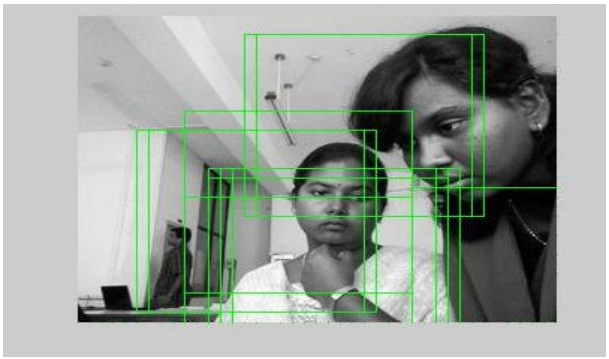

Figure 13. Video –Two Face Image.

IV. CONCLUSION

In this work, the face object is detected for security and surveillance based application using Adaptive Boost algorithm. Rectangle sum and integral image is calculated for every training image and input images using Haar-like feature. Thus the face object is detected for various size of still images and video using the Adaptive Boost algorithm.

ACKNOWLEDGMENT

The authors extend gratitude to the parent institution where the authors were working for all support and facilities provided for the successful completion of the work.

REFERENCES

[1] F. Freund and R. E. Schapire, "A short introduction to boosting," *J. Japan. Soc. for Artif. Intel.*, vol. 14, pp. 771-780, Sep.1999.

[2] R. E. Schapire, "The boosting approach to machine learning: An overview," *in MSI Workshop Nonlinear Estimation Classification*., 2002, pp. 1134-1227

[3] C. C. Kyrkou and T. Theocharides, "A flexible parallel hardware architecture for adaboost-based real-time object detection," *IEEE Trans. On Very Large Scale Integration System*, vol. 19, June 2011.

[4] Training face database [Online]. Available: http://cbcl.mit.edu./software-datasets/FaceData2.html

[5] T. Theocharides, N. VijayKrishnan, and M. J. Irwin, "A parallel achitecture for hardware face detection," in *Proc. IEEE Comput. Soc. Annu. Symp. Vlsi Des (ISVLSI), Karlsruhe*, Germany, pp. 452-453

[6] P. Viola and M. Jones, "Real-time object detection," *Int. J.Comput. Vision*, vol. 55, pp. 137-154, 2004.

[7] P. Viola and M. Jones, "Rapid object detection using boosted cascade of simple feature," *Accepted Conf. of Comput.Vision and Pattern Recognition*.

[8] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proc. IEEE ,Cambrige*, U.S.A, 1997, pp. 130-137

N.Ramya received her B.E Degree in Electronics and Communication Engineering from Anna University Chennai, India in 2010.She received her M.E Degree in Applied Electronics from Anna University Chennai, India in 2012. She is currently Working as an Assistant Professor in Jerusalem College of Engineering, Chennai, India. Her current research interest includes Image Processing.

G.Durga received the B.E degree in Electronics and Communication Engineering from University of Madras, India in 2002 and the M.E degree in Applied Electronics from Anna University, India in 2004. She is currently working toward the Ph.D degree in the area of Nano Electronics, Anna University, India. Since 2005, she has been a.faculty member in the Department of Electronics and Communication Engineering, SSNCE, Tamilnadu, India. Her research interests are in the area of Nano Electronics, Semiconductor Device Modeling and Analysis and Low power VLSI. She has published research papers in International Journal and Conferences