REAL TIME HUMAN FACE DETECTION AND TRACKING

Jatin Chatrath[#1], Pankaj Gupta[#2], Puneet Ahuja[#3] , Aryan Goel[#4], Shaifali M.Arora[*5]

[#] *B.Tech, Electronics and Communication, MSIT (GGSIPU)*
C-4 Janak Puri NEW DELHI-58, INDIA
[1]jatinchatrath@gmail.com

[*] *M.Tech , Electronics and Communication, asst professor at MSIT (GGSIPU)*
C-4 Janak Puri NEW DELHI-58, INDIA
[2]shaifali04@yahoo.co.in

***Abstract*-- This paper describes the technique for real time human face detection and tracking using a modified version of the algorithm suggested by Paul viola and Michael Jones. The paper starts with the introduction to human face detection and tracking, followed by apprehension of the Vila Jones algorithm and then discussing about the implementation in real video applications. Viola jones algorithm was based on object detection by extracting some specific features from the image. We used the same approach for real time human face detection and tracking. Simulation results of this developed algorithm shows the Real time human face detection and tracking supporting up to 50 human faces. This algorithm computes data and produce results in just a mere fraction of seconds.**

Keywords— Human face detection, Integral Image, AdaBoost.

I. INTRODUCTION

With increasing terrorist activities and augmenting demand for video surveillance, it was the need of an hour to come up with an efficient and fast detection and tracking algorithm. Many real time face tracking systems have been developed [2][3][4][5][6] in the past. In this paper, we proposed a more efficient algorithm that consists of three intermediate steps, first is the development of a new image representation called "*integral image"* [8], which allows feature selection to be easy and rapid. Second step deals with the construction of classifiers that helps us to segregate desired features from the set of large number of features using a technique called *"AdaBoost"* [7]. Third step deals with the cascading of different classifiers which was introduced in step 2 for further detailed selection of features and thereby narrowing down our search and increasing speed of detection and tracking.

This detecting and tracking algorithm will bring practical applications like Smart captcha, Webcam based energy/power saver, Time tracking service, Outdoor surveillance camera service, Video chat service, Teleconferencing.

In this paper, Section 2 describes the types of features for human face detection. Section 3 describes the algorithm for human face detection. Section 4 describes the efficient algorithm and flowchart for detection and tracking. Section 5 and 6 describes the simulation results and conclusion respectively.

II. FEATURES FOR FACE DETECTION

Images are generally classified based on the value of simple features. It is advantageous to use features rather than using pixels as *feature based systems* operates much faster than *pixel based systems*. Ad-hoc domain knowledge which is very difficult to learn using a finite quantity of training data can be encoded using features.

Generally, we use three types of features for face detection procedure, namely *two-rectangle* features, *three-rectangle* feature and *four-rectangle* feature. *Two-rectangle feature* is the difference between the sums of the pixels within two rectangular regions. The regions are of same size and shape and are horizontally or vertically adjacent, as shown in figure 1(C, D). A *three-rectangle feature* calculates the sum within two outside rectangles subtracted from the sum in a center rectangle, as shown in figure 1(B). Finally a *four-rectangle feature* computes the difference between diagonal pairs of rectangle, as shown in figure 1(A).

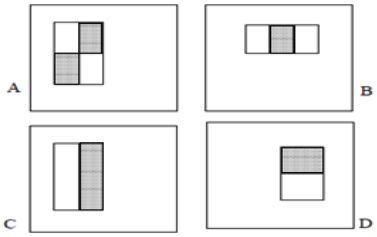

Fig 1: (A) shows four-rectangle feature, (B) shows three-rectangle feature, while (C) and (D) shows two-rectangle features.

III. RAPID HUMAN FACE DETECTION

Algorithm used for face detection system consists of three major steps as discussed below.

3.1. Integral Image Formation

Rectangle features can be computed very rapidly using an intermediate representation for the image which we call the integral image [11]. The formula for calculation of integral image is as follows. The integral image at location x,y contains the sum of the pixels above and to the left of x,y.

$$jj(x,y) = \sum_{x' \le x; y' \le y} j(x',y') \quad (1)$$

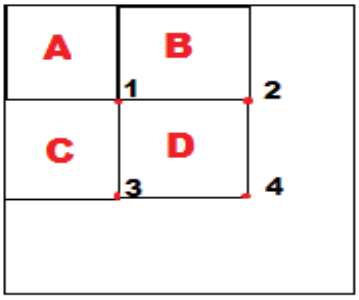

Fig 2: Integral image at location 1 is the sum of pixels in region A; at location 2 sum of pixels in region A+B, at location 3 C+A and at location 4 A+B+C+D

Where jj(x,y) is an integral image and j(x',y') is the original image. Using the following formula,

$$r(x,y) = r(x,y-1) + j(x,y) \quad (2)$$
$$jj(x,y) = jj(x-1,y) + r(x,y) \quad (3)$$

(Where *r(x,y)* is the cumulative row sum *r(x,-1) = 0* and *jj(-1,y)=0)* the integral image can be computed from one pass over the image.

3.2 AdaBoost

AdaBoost is a learning classification function which when given a set of features and a training set of positive and negative images, to learn a classification function any number of machine learning approaches could be used. AdaBoost is used to train the classifiers as well as to select a small set of features [7]. When used in its real form, AdaBoost learning function boosts the performance of a simple learning algorithm (sometimes called weak). AdaBoost has capability to achieve large margins rapidly which is one of the key features of this algorithm.

Each sub window consists of 180,000 features associated with it which is much larger than the number of pixels. Features can be computed very efficiently but computing the complete set is expensive. Minuscule number of these features can be combined to form an efficient classifier. Most importantly the challenge is to find these features. A weak learning algorithm is designed for development of a classifier that learns itself, and adjusts accordingly by changing its threshold value. Let $c_j(x)$ denotes the classifier, f_j denotes feature, φ_j denotes threshold and a parity p_j indicating the direction of inequality where x is a 24x24 pixel sub-window of an image.

$$c_j(x) = \begin{cases} 1 & if\ p_j f_j < p_j \varphi_j \\ 0 & otherwise \end{cases} \quad (4)$$

The feature selected in early rounds for initial classifiers having less complexity has an error rate between 0.1 to 0.3. Features selected in later rounds for complex classifiers the error rate reaches between 0.4 and 0.5.

AdaBoost generates and calls a new classifier in each of a series of round t=1,….,T. For each call a distribution of weights w_t is updated that indicates the importance of images in the data set for the classification. On each round the weights of each incorrectly classified image are increased and the weights of each correctly classified image are decreased, so the new classifier focuses on the images which have so far eluded correct classification. The procedure of boosting is as follows:

- Given example images(x_1,y_1) …….(x_n,y_n) where $y_i = 0$ for negative and $y_i = 1$ for positive examples.
- Initialize weights $w_{1,i} = \frac{1}{2u}, \frac{1}{2v}$ for $y_i = 0,1$ respectively where u and v are the number of negatives and positive respectively.
- For t=1,……..,T:
    1. Normalize the weights,
    $$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \quad (5)$$
    so that w_t is a probability distribution.

2. For each feature, j, train a classifier t_j which is restricted to using a single feature. The error is evaluated with respect to $w_{t,}$
$\xi_j = \sum_i w_i | c_j(x_j) - y_j |$. (6)
3. Choose the classifier, $t_{t,}$ with the lowest error ξ_t .
4. Update the weights:
$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$ (7)
where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\xi_t}{1-e_t}$.

- The final strong classifier is:

$$c(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t c_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & otherwise \end{cases} \quad (8)$$

where $\alpha_t = \log \frac{1}{\beta_t}$. (9)

This is the AdaBoost algorithm for classifier learning. After every round of boosting each feature is selected from 180,000 potential features [8, 9, 10].

It requires 0.7 seconds to scan through a 384 by 288 pixel image. But to increase number of features directly to classifiers, increases the computation time.

Fig 3: The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

Initial features selected by AdaBoost is very easy to interpret and meaningful for face detection. Most important feature is that the region of the eyes is darker than cheeks, as shown in figure 3. The bridge between the eyes is lighter than the eyes shade.

3.4 Cascading

This section discuss in detail about the cascading of classifiers and their training whose training functions were defined in the previous section. Increase performance detection while reducing the detection time are the key features of this algorithm. Simpler classifiers separate the positive and the negative from the image, hence are called weak classifiers or initial classifiers. Then additional features are added to other classifiers and hence are called complex or strong classifiers. They are so called because their complexity goes on increasing as we go towards the final output, they take the output of the previous simpler classifier as their input and perform either of the two operations, i.e. they either reject the window, or pass it on and trigger the next more complex classifier for further processing. This process keeps on going until the accurate detection is achieved otherwise the sub window is rejected in the path. The sub window reaching the output has to pass all the intermediate classifiers in order to get detected [14]. This overall form of detection is that of a degenerate decision tree, what we call a "cascade", as shown in figure 4.

Stages in the cascade are constructed by AdaBoost using training classifiers and then adjusting its threshold to minimize false negatives. Default AdaBoost threshold's value is designed to yield low error rates or training data. In general a lower threshold yields higher detection and higher false positive rates.

For reducing computation time we need to reduce the number of features. In the classifiers, but this decreases the accuracy. Hence there is a tradeoff between the accuracy and speed [15]. In order to overcome this problem, first stage classifier can be constructed from a two feature strong classifier reducing the threshold to minimize false negative. Adjustment of the threshold can be done to detect 100% of the faces with a false positive rate of 40%. See figure 3 for the description of two features used in the classifier.

As a result, the second classifier goes through more difficult task then the first. The complete motto is to increase the probability of face detection by eliminating unwanted areas in early stages only. In further stages more specific and detailed elimination is done thereby further increasing the probability [12, 13, 15].

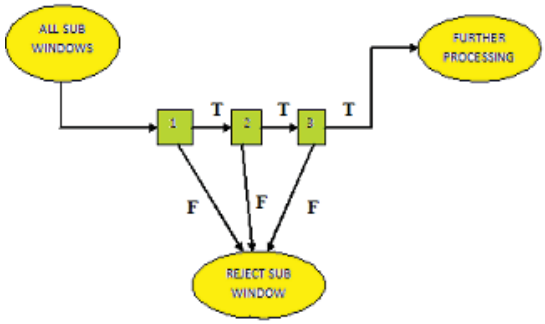

Fig 4: Every sub window is subjected to a series of classifiers in which early stages eliminate the most of the negative examples with very little processing, detailed and specific elimination is done by subsequent stages. To do this they require more complex processing.

IV. IMPLEMENTATION OF REAL TIME FACE DETECTION AND TRACKING ALGORITHM

In Matlab 2013 version, we have used computer system Vision toolbox's function called vision.CascadeObjectDetector which uses the Viola-Jones algorithm to detect people's faces, noses, eyes, mouth, or upper body. We can also use the trainCascadeObjectDetector function if we want to train a custom classifier to use with this System object.

We carried out our demonstration on dell, i5 third generation processor and found image detection to work profoundly. We developed the code for real time detection and tracking of human faces. For this we used imaqhwinfo command in Matlab 2013 to get information about all the adapters installed in our laptop.

```
>> imaqhwinfo

ans =

    InstalledAdaptors: {'gentl'  'gige'  'matrox'  'winvideo'}
       MATLABVersion: '8.1 (R2013a)'
         ToolboxName: 'Image Acquisition Toolbox'
      ToolboxVersion: '4.5 (R2013a)'
```

As we were working on windows operating system, we selected 'winvideo' as our default adapter. Then we configured the camera according to our needs through following instructions.

```
vid=videoinput('winvideo',1,'YUY2_640x480');
vid.returnedcolorspace='rgb';
vid.framespertrigger=1;
vid.TriggerRepeat=inf;
triggerconfig((vid),'manual');
```

We set the output resolution to 640×480 pixels and return color space as 'rgb'. Frames per trigger were set to 1 and trigger repeat was set to infinity. We changed the default triggering method to 'manual'.

Then we started the camera by using start () function. We looped the frames acquiring function till 100 frames and triggered the camera object. Then we used 'imread' to acquire the image and then 'imshow' method to display it frame by frame on the console. Shape inserter system object is used to draw the rectangle around the detected face in real time. The box moves in accordance with the real time movement of the subject's face.

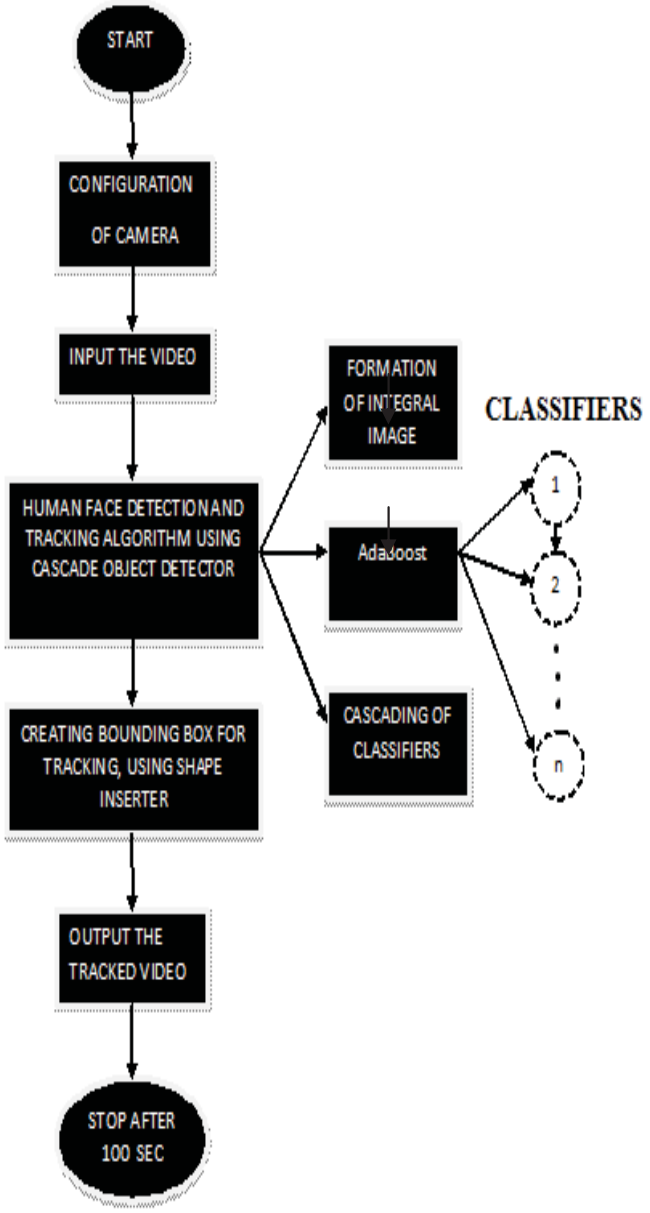

Fig 5: Flowchart for real time detection and tracking algorithm.

V. SIMULATION RESULT

We have presented an approach for face detection and tracking with minimization in the computation time while achieving high detection and tracking accuracy. We have successfully developed a real time human face detection algorithm better than the previous versions in terms of computation time and feature selection. We run our code in i5 processor dell laptop and the results were in accordance with the aim of this research.

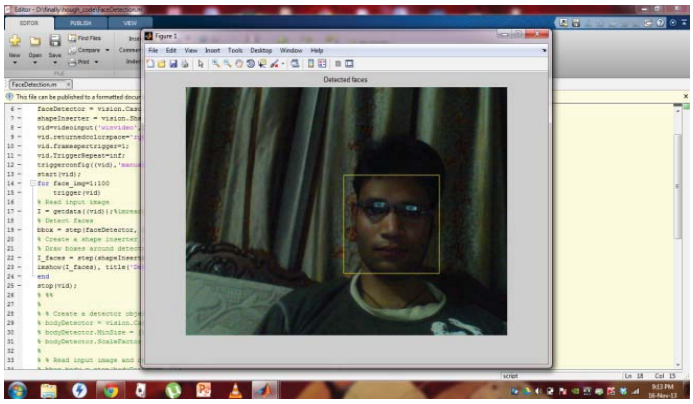

Figure 5: Real time face detection and tracking of one of the authors of this research paper

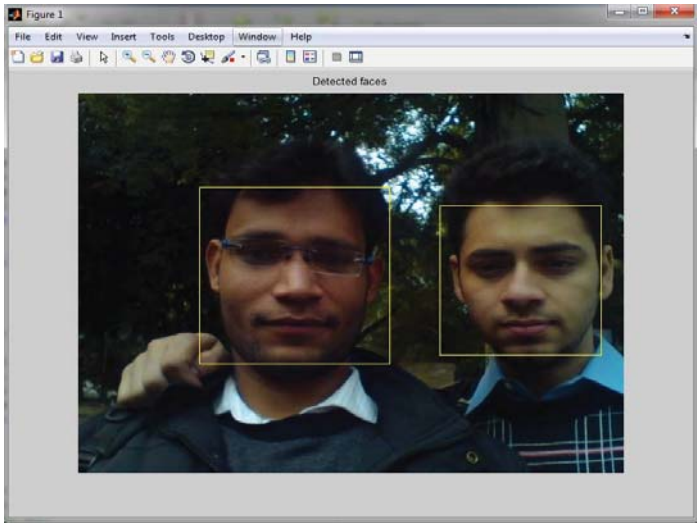

Figure 6: Multiple face detection and tracking.

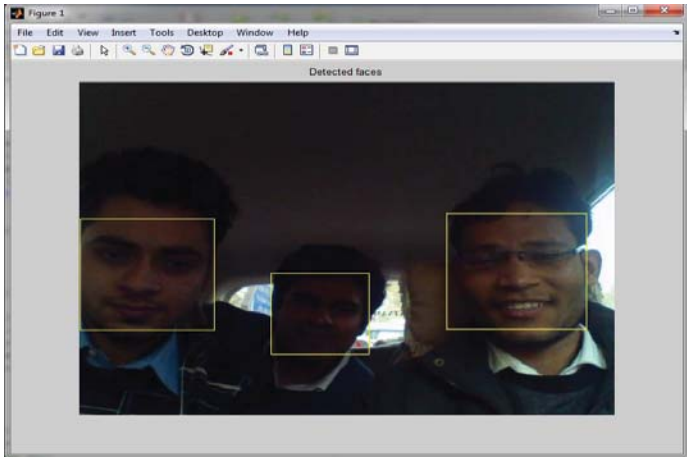

Figure 7: Multiple face detection and tracking.

VI. CONCLUSION

This paper brings altogether a new algorithm, representations and insights which are generic and may well have broader application in computer region and image processing. Finally, this paper presents a set of detailed experiments on difficult face detection and tracking data set which has been widely studied. This data set includes faces under a wide range of conditions including: illumination, scale, pose and camera variation. Nevertheless the system which work under this algorithm are subjected to same set of conditions and but the algorithm is flexible enough to adjust according to the changing conditions.

REFERENCES

[1]http://www.cse.ohiostate.edu/otcbvs/05/OTCBVS-05-FINALPAPERS /W01_16.pdf.

[2] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland,"Pfinder: Real-time tracking of the human body," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.19, 1997, pp. 780-785.

[3] I.Haritaoglu, D.Harwood and L.S.Davis ,"W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People", In Proc. Of the International Conference on Face and Gesture Recognition, April, 1998

[4] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking Groups of people.Computer Vision and Image Understanding. 80:42-56, 2000

[5] J. Connell, A.W. Senior, A. Hampapur, Y-L Tian, L. Brown, and S. Pankanti, "Detection and Tracking in the IBM PeopleVision System", IEEE ICME, June 2004.

[6] L.M.Fuentes and S.A.Velastin, "People Tracking in surveillance application", in Proc. 2nd IEEE International Workshop on PETS, Dec. 2001

[7] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Eurocolt '95*, pages 23–37. Springer-Verlag, 1995.

[8] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *International Conference on Computer Vision*, 1998.

[9] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[10] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann. Stat.*, 26(5):1651–1686, 1998

[11] Rapid Object Detection using a Boosted Cascade of Simple Features, by Paul Viola Michael Jones. Accepted conference on computer vision and pattern recognition 2001.

[12] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

[13] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *International Conference on Computer Vision*, 2008

[14] K. Sung and T. Poggio. Example-based learning for viewbased face detection. In *IEEE Patt. Anal. Mach. Intell.*, volume 20, pages 39–51, 1998.

[15] J.K. Tsotsos, S.M. Culhane,W.Y.K.Wai, Y.H. Lai, N. Davis, and F. Nuflo. Modeling visual-attention via selective tuning. *Artificial Intelligence Journal*, 78(1-2):507–545, October 1995.