INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

FACE DETECTION AND RECOGNITION IN REAL TIME VIDEO SURVEILLANCE

Akintola K.G., Akinyokun O.C., Olabode O.

Computer Science Department, Federal University of Technology Akure, Ondo State, Nigeria

ABSTRACT

Emerging smart visual surveillance systems are requiring automatic detection and recognition of human beings within the scene and the prediction of the actions being performed by the detected human objects. These are challenging issues especially in an unconstrained environment. This paper presents a framework for the automatic detection and recognition of human beings from video cameras via smart visual systems that automatically sense and correctly recognize human identity and actions by means of Machine vision techniques. Such systems require low response time in terms of image processing and acceptable recognition accuracy. Initial human detection is addressed by background subtraction techniques using parallel–processed Kernel Density Estimation (PKDE). Temporal tracking of the objects' trajectories is performed by employing a spatial body tracking system designed as a multi-part colour histogram-based tracker. In face recognition, the Principal Component Analysis (PCA) algorithm was implemented. An experiment was performed in computer laboratory of Federal University of Technology Akure where a camera was installed in the Laboratory to capture students entering the lab. The face detection algorithm performs well and reduces the computational time. The detector is 2.5 times as fast as Viola and Jones method, although there were false positive faces detected. Some future areas of practical application of such system include access control to facilities like lecture rooms, Automated Teller machines, and attendant management systems.

Keywords: PCA, KDE, A Haar-like feature

1.0 Introduction

In today's self-service world, the need for securing physical properties and assets is becoming increasingly important. Recently technology became available to allow verification of the true identity of criminals. This technology is based on a field called Biometrics. Biometric access control is automated methods of verifying or recognizing the identity of a living person on the basis of some physiological and behavioral characteristics. Recognizing faces in videos is a fundamental task for realizing surveillance systems or intelligent vision-based systems for human monitoring, identity recognition and activity analysis. To be able to recognize humans, in a surveillance scenario, robust, efficient and fast face detection and recognition algorithms are required. Viola and Jones proposed the use of the Haar-like features for face detection. Viola and Jones also proposed Adaboost algorithm for constructing a strong classifier used to select efficient features for classification. These algorithms have been found very efficient at face detection. Viola and Jones algorithms combined with motion information is proposed in this work for fast face detection. The Principal Component Analysis (PCA) is used for face recognition. The Haar-like features can be computed at any scale or location in constant time using the integral representation of an image.

This paper describes a solution for human identification using video signals which was specially designed for use in facility access control. Towards this end it was coined a face detection and recognition in real time video

surveillance with a preprocessing stage based on a rapid frontal face detection system using Haar-like features introduced by Viola et al.,(2001). The face recognition system is based on the eigenfaces method introduced by Turk et al.,(1991). Eigenvector-based methods are used to extract low-dimensional subspaces which tend to simplify tasks such as classification. The system that will be used in access control to facilities is able to robustly detect and recognize faces at approximately 16 frames per second in a 1GHz Pentium III laptop.

PCA is one of the most primitive algorithms used for face recognition proposed by Turk etal., 1991. Eigenfaces method is the implementation of Principal Component Analysis (PCA) over images. The Eigenfaces method tries to find a lower dimensional space for the representation of the face images by eliminating the variance due to non-face images. In this method, the features of the studied images are obtained by looking for the maximum deviation of each image from the mean image. This variance is obtained by getting the eigenvectors of the covariance matrix of all the images.

2.0 Related Work on Face Detection and Recognition

Face recognition focuses on recognizing the identity of a person from a database of known individuals. Face recognition has several advantages over other biometric technologies; it is natural, non – intrusive and easy to use (Jain, 2004). There are two predominant approaches to the face recognition problem: Geometric (feature based) and Holistic based methods. Caetano *etal.,* (2001) proposed a Probabilistic Model for human skin color detection in videos. The skin color is modeled in the chromatic subspace using multivariate statistical which is by default normalized with respect to illumination. The motivation for this is to perform automatic human face recognition in video scenes using color intensity values and mixture of Gaussians models. The skin images is modeled using multivariate statistics and the model is used to segment skin images from the rest of the scene.

Kanade *etal.,* (1973) first proposed a Neural Network-based (NN) approach to facial recognition. Although NN have received significant attention in many research areas, few applications were successful in face recognition because of the following reasons

a. It is easy to train a neural network with samples which contain faces, but it is much harder to train a neural network with samples which do not.
b. The numbers of "non-face" samples are unavoidably just too large in practice.

Brunelli etal., (1993) proposed geometric feature based approach to facial recognition. The objective is to recognize face in images using geometric and template matching. Two algorithms for face recognition: geometric feature based matching and template matching were developed. The geometric feature based matching approach extracts 35 facial features automatically such as eyebrow thickness and vertical position, nose vertical position and width, chin shape and zygomatic breadth. These features form a 35-D vector and recognition is performed using a Bayes classifier. The limitation of this approach is the difficulty in getting the facial features. Template matching methods such as Brunelli etal., (1993) operates by performing direct correlation of image segments (e.g. by computing the Euclidean distance). Template matching is only effective when the query images have the same scale, orientation, and illumination as the training images (Cox et al., 1995).

Rowley etal, (1998) proposed Neural Network model (NN) for face detection. The approach can detect faces at multiple scales. The image window is first preprocessed and then given to neural network to detect facial features in the window. The networks have three types of hidden units: 4 for 10x10 pixel sub-regions, 16 for 5x5 pixel sub-regions and 6 for 20x5 pixel sub-regions. These sub-regions are chosen to represent facial features that are important to face detection. Overlapping detections are merged.

Kadoury et al., (2006) presents "Face Detection in grey scale images using locally Linear Embedding". It involves mapping face and non-face data to LLE and then using support vector machines to classify face and non-face images. The LLE method performs dimensionality reduction on data for learning and classification purposes. Proposed by Roweis and Saul (2000), the intent of LLE is to determine a locally linear fit so that each data point can be represented by a linear combination of its closest neighbors. The research first applied the LLE algorithm to 2D facial images to obtain their representation in a sub-space. The low-dimensional data are then used to train support vector machine (SVM) classifiers to label windows in images as being either face or non-face. Six different databases of cropped facial images, corresponding to variations in head rotation, illumination, facial expression, occlusion and aging, were used to train and test the classifiers. Experimental results obtained demonstrated that the performance of the proposed method was better than other face detection methods, thus indicating a viable and accurate technique.

Viola and Jones (2001) presents fast object detection using Haar-like features and a cascade of classifiers. Their algorithm has been adjudged the best face detection algorithm recently, therefore this we adopted this algorithm combined with our motion algorithm for fast face detection.

3.0 Kernel Density Estimation for background subtraction

Kernel density estimation (KDE) is the most used and studied nonparametric density estimation method. The model is the reference dataset, containing the reference points indexed natural numbered. In addition, assume a local kernel function centered upon each reference point, and its scale parameter (the bandwidth). The common choices for kernels include the Gaussian: and the Epanechnikov kernel (Elgammal etal., 1991)

The Gaussian Kernel is given by:

$$K_N = (2\pi)^{-\frac{d}{2}} \exp\left(-\tfrac{1}{2}\|x\|^2\right) \quad (1)$$

The Epanechnikov kernel is given by:

$$K_E = \begin{cases} -\frac{1}{2}c_d^{-1}(d+2)(1-||x||^2 & if\ ||x|| < 1 \\ 0 & otherwise \end{cases} \quad (2)$$

Let $x_1, x_2, \ldots \ldots x_n$, be a random sample taken from a continuous, univariate density *f*. The kernel density estimator is given by,

$$\widehat{f}(x;h) = \frac{1}{nh}\sum_{i=1}^{n} k\left(\frac{x-x_i}{h}\right) \quad (3)$$

K is the function satisfying $\int k(x)dx = 1$ which is refered to as the Kernel.
h is a positive number, usually called the bandwidth or window width.

3.1 Histogram computation.

The first 100 initial frames in the video sequence (called learning frames) are used to build stable distributions of the pixel RGB mean. The RGB intensities of each pixel position is accumulated for the 100 frames and we calculate the cumulative sum of the average intensities i.e (sum of (RGB)/3) is computed over 100 frames. A histogram of 256 bins is constructed using these pixel average intensities over the training frames. . The sum is then normalized to 1. That is we divide each histogram bin value with the accumulated sum to get a normalized histogram as shown in figure 1.

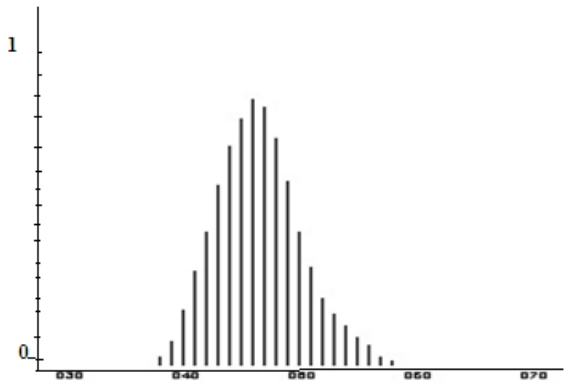


Figure 1. An Histogram of typical pixel location.

3.2 Threshold calculation.

Threshold is a measure of the minimum portion of the data that should be accounted for by the background.
For more accuracy in our segmentation, we use different threshold for each histogram bins.
The pseudo- code for the Threshold calculation is given below

```
1 For each H[i]
2       Get sum of H[i]
3       Peak[i]=max(H[i])
4       Pth[i]=Peak[i]/2
```

```
5       Calculate sum2(H[i] > Pth[i])
6           If(sum2(H[i] > Pth[i])  is less than 0.95 of sum of Hi
7               Pthi=Peak[i]/2
8               go to 5
9           else
10              threshold=Pth[i]
```

3.3 Foreground/ Background detection.

For every pixel observation, classification involves determining if it belongs to the background or the foreground. The first few initial frames in the video sequence (called learning frames) are used to build histogram of distributions of the pixel means. No classification is done for these learning frames. Classification is done for subsequent frames using the process given below. Typically, in a video sequence involving moving objects, at a particular spatial pixel position a majority of the pixel observations would correspond to the background. Therefore,

background clusters would typically account for much more observations than the foreground clusters. This means that the probability of any background pixel would be higher than that of a foreground pixel. The pixel are ordered based on their corresponding value of the histogram bin.

The Background detection Algorithm

(Frames 1—N is used for training modeling the background).
Read frames 1 -- N
For each pixel
Calculate the value of (r+g+b)/3
Locate the corresponding bin value in the histogram of the pixel.
Increment the bin of this value by 1
Increment the surrounding bandwidth pixels by fraction of 1
Normalize the histogram value by dividing each bin value by the sum of bins.
Calculate the adaptive threshold as given in figure xx.
Read the Next frame after N.
For each pixel
Read the intensity of RGB of the pixel.
Calculate the value of (r+g+b)/3
Locate the corresponding bin value in the histogram of the pixel.
Test if the value is < threshold
Classify the pixel as foreground
Else
Classify the pixel as background

4.0 Viola – Jones AdaBoost face detector

Viola and Jones (2001) presents fast object detection using Haar-like features and a cascade of classifiers. The Viola-Jones Detector consists of three parts: the first part is concerned with encoding the image data using integral image for rapid computation of Haar-like features that can form a template to model human face variation. The second part is the use of AdaBoost Algorithm for selection of efficient classifiers from a large population of potential classifiers. The third part is the efficient method of combining the classifiers generated by AdaBoost algorithm into a cascade, which can remove most of the non-face images in the early stage by simple processing by focusing on complex face images in the later stages which requires higher processing time.

The Algorithm

a. Integral Image and Haar-like Features
b. AdaBoost Algorithm
c. Cascade of Classifiers

a. Integral Image and Haar-like Features

AdaBoost algorithm classifies images based on the value of simple features. The simple features are similar to Haar basis function as shown in figure 2.0. In the diagram, we have three kinds of features: two two-rectangular features, one three-rectangular features, finally one four-rectangular features. The feature value is the difference between the sums of pixels of the white region to the dark region of the features. Haar-like features have scalar values that represent differences in average intensities between two rectangular regions. They capture the intensity gradient at different locations, spatial frequencies and directions by changing the position, size, shape and arrangement of rectangular regions exhaustively according to the base resolution of the detector. For

example, when the resolution is 19 x 19 pixels, 80,160 features are generated from the feature sets (a) to (d) in Figure 2. A weak learning algorithm is designed to select the single feature that best separates the face and nonface examples. A small number of effective features are selected by updating the sample distribution using AdaBoost.

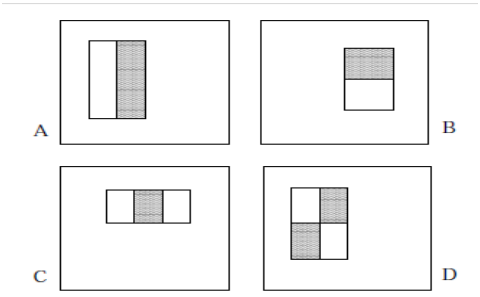


Figure 2 Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

The weak classification function select a single rectangular feature which best separates the positive examples from the negative examples.

$$h_j(x) = \begin{cases} 1 & if \quad \rho_j . f_j(x) > \rho_j . \theta_j \\ 0 & otherwise \end{cases} \qquad (4)$$

Where θ_j is a threshold and ρ_j is a parity indicating the direction of the inequality sign. The value of θ_j and ρ_j are determined so that the error rate is minimized.

Computing the features involves starting from every possible pixel of the detector sub-window and also should cover all the widths and heights possible (all the rectangles possible).

The concept of integral image is very simple. You preprocess the image to significantly increase the extraction of Haar-like features for analysis and object detection. At any point (i, j) in the original image, you sum up all the pixels to the left and up from that point (i, j): I(x) = sum I(i, j). After that point, to get the sum of all pixel values inside the **S** rectangle, we need only four array references: S = A - B - C + D, where A, B, C, D are the points in the integral image. See figure 3.

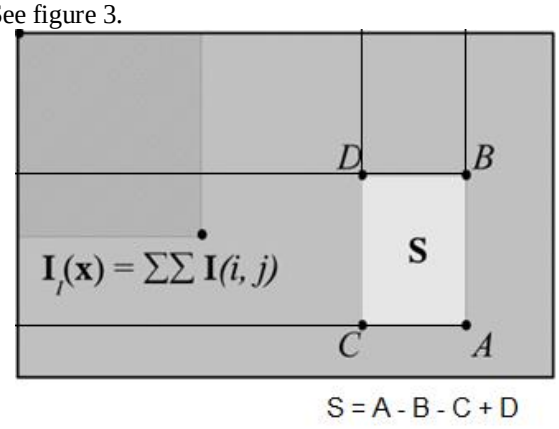


Figure 3 Integral Image calculations

The features consist of boxes of different sizes and locations. Consider some 20x20 rectangle; you may place, for example, inside it two rectangles of size 10x20 or four rectangles with size 10x10. Having such a feature basis of 20x20 rectangular features, you project the image to that set. Keeping in mind that you have the integral image, such a projection step takes a small amount of time. For a feature consisting of two rectangles of 10x20 size, one needs to compute the sum of all the pixels in that 10x20 rectangles as was pointed in the previous section, so 4*2 = 8 array references, instead of an ordinary floating point matrix multiplication taking 2*20*20 = 800 operations is required.

b. AdaBoost Algorithm

In a single sub-window of 19 X 19 pixels, we can generate more than 80,160 features. Using all these features during detection can consume a lot of time. But there are efficient classifier among these large number of features which when efficiently combined, give a better classifier. Viola and Jones gave a variant of AdaBoost algorithm to select efficient classifiers. It works by combining a set of weak classifiers to form a strong classifier. AdaBoost finds the new weak classifier after re-weighting the training examples such that incorrectly classified examples get more weight. The weak classification function select a single rectangular feature which best separates the positive examples from the negative examples using equation (4).

The boosting Algorithm is as follow:

A set of N labeled training examples is given as $(x_i, y_i), \ldots, (x_N, y_N), where\ y_i \in \{0,1\}$ is the class label associated with example x_i. $w_{t,i}$ is a weight of example x_i. The weight are initialized by $w_{t,i} = \frac{1}{2m}, \frac{1}{2l}$ where m = number of negative examples and l = number of positive examples. The final strong classifier $H(x)$ is a linear combination of T weak classifiers $h_t(x)$:

$$H(x) = \begin{cases} 1 & if (\sum_{t=1}^{T} \propto_t h_t(x)) \geq 1/2 (\sum_{t=1}^{T} \propto_t) \\ 0 & otherwise \end{cases} \quad (5)$$

In each round of the boosting process, the best joint Haar-like feature is selected according to the step (A) to (E)

1. Given example images $(x_i, y_i), \ldots, (x_N, y_N), where\ y_i \in \{0,1\}$ for face and nonface examples respectively.
2. Initialize weights $w_{t,i} = \frac{1}{2m}, \frac{1}{2l}$ for y_i =0,1 respectively where m = number of negative examples and l = number of positive examples
3. For $t = 1, \ldots, T$:
   a. For each feature, calculate a feature value.

      Normalize the weights

      $$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}. \quad (6)$$

   b. Train a weak classifier h_j using each feature. The weak classifier can only use a single feature.

      The error is evaluated with respect to $w_{t,i}$,

      $$\epsilon_t = \sum_{i: w_{t,i}} | h_j(x_i) - y_i | \quad (7)$$

   c. Choose the classifier $h_j(x)$ with the lowest error ϵ_t
   d. Update the weights:

      $$w_{t+1 = w_{t,i}} \beta_{t,i}^{1-\epsilon_t} \quad (8)$$

4. The final strong classifier is:

$$H(x) = \begin{cases} 1 & if (\sum_{t=1}^{T} \propto_t h_t(x)) \geq 1/2 (\sum_{t=1}^{T} \propto_t) \\ 0 & otherwise \end{cases} .. \quad (9)$$

$$\text{Where } \propto_t = \log\left(\frac{1}{\beta_t}\right) \quad (10)$$

The final strong classifier is created by combining the weak classifiers generated and setting the threshold as half the weight given to the classifiers.

c. Cascade of classifiers

This section describes an algorithm for constructing a cascade of classifiers (Viola and Jones, 2001) which achieves increased detection performance while radically reducing computational time. The key insight is that smaller, and therefore more efficient, boosted classifiers can be constructed which reject many of the negative sub-windows while detecting almost all positive instances. Simpler classifiers are used to reject the majority of sub-windows before more complex classifiers are called upon to achieve low false positive rates.

A cascade of classifiers is degenerated decision tree where at each stage a classifier is trained to detect almost all objects of interest while rejecting a certain fraction of the non-object patterns (Viola and Jones, 2001).
Each stage (strong classifier) was trained using the Adaboost algorithm. At each round of boosting is added the feature-based classifier that best classifies the weighted training samples. With increasing stage number, the number of weak classifiers, which are needed to achieve the desired false alarm rate at the given hit rate, increases.
For the training of the cascade of classifiers, each stage (strong classifier) is generated by using AdaBoost algorithm. To achieve high detection rate and low false positive rate of the final detector, each stage is made such that it has the detection rate greater or equal to a given value and false positive rate less than or equal to a given value. If this condition is not met, the stage is again trained by specifying the larger number of classifiers than the previous. The final false positive rate is given as:

$$F = \pi_{i=1}^{k} f_i \quad (11)$$

Where F is the false positive rate of the cascaded classifier. f_i is the false positive rate of the ith stage and k is the number of stages. The detection rate is

$$D = \pi_{i=1}^{k} d_i \quad (12)$$

Where D is the detection rate of the cascaded classifier. d_i is the detection rate of the ith stage and is the number of stages. After the generation of one stage, the non-faces images for the next stage are obtained from the false positives of the previous non-face images. The training algorithm is as given below:

a. The trainer supplies values of f, the maximum acceptable false positive rate per layer and d, the minimum acceptable detection rate per layer.
b. The trainer supplies target overall false positive rate, F_{target}
c. P= set of positive examples
d. N=set of negative examples
e. $F_o = 1.0;\ D_o = 1.0,$
f. i = 0
g. while $F_i > F_{target}$
i =i+1
n_i=0; $F_i = F_{i-1}$
while $F_i > f * F_{i-1}$
$n_i = n_i$+1

Use P and N to train a classifier with n_i features using AdaBoost. Evaluate current cascade classifier on validation set to determine F_i and D_i. Decrease threshold for the ith classifier until the current cascade classifier has a detection rate of at least d*D_{i-1} this also affects F_i.
Set N=ϕ
If $F_i > F_{target}$, then evaluate the current cascade detector on the set of non-face images and put any false detections into the set N (to be used by the next stage)

This algorithm continues until the final positive rate is less than or equal to the target false positive rate. For more detail see (Viola and Jones, 2001)).
The limitation of Viola and Jones (2001) is that the training time is too long and it is also computationally expensive.

5.0 A spatio-color Histogram Algorithm for Scalable Human Object Tracking

The proposed algorithm is composed of two stages. First is the appearance correspondence mechanism. Once detected, Appearance models are generated for objects appearing in the scene. The model is the estimate of probability distribution of colour of pixel colours. Multiple models are developed for a single object. These models are then used in subsequent frames to match the set of currently detected models and that of target models. In the second phase occlusion and object merge and separation are handled. The foreground object detected in previous stage is passed to the object tracker. This information is the appearance model of the object.
We adopt a multi-part tracking algorithm in our system. That is, we segment each silhouette into upper-body area and lower-body area and generate a histogram of colures in HSV color space for each region. This approach is good enough at discriminating individuals because of varying intensity in identical objects with similar color and occlusion. Our approach makes use of the object color histograms of previous frame to establish a matching between objects in consecutive frame. Our method is also able to detect object occlusion, object separation and label the object appropriately during and after occlusion.

6.0 Face Recognition Using Principal Component Analysis (PCA)

The Eigenface space is obtained by applying the eigenface method to the training images. Later, the training images are projected into the Eigenface space. Next, the test image is projected into this new space and the train image projection with the minimum distance from the test image projection is the correct match for that test face.

6.1 Training Operation in PCA

Let I be an image of size(Zx , Zy) pixels, then the training operation of PCA algorithm can be expressed in mathematical terms as:

Convert the training image matrix I of size (Nx, Ny) pixels to the image vector Γ of size (P×1) where P = Zx × Zy (i.e: the train image vector Γ is constructed by stacking each column of train image matrix I)

Create a training set of training image vectors such that its size is P × M where M is the number of training images.

Γ p× M = {Γ_1 Γ_2….Γ M} where, Γ_i represents the image vector of i^{th} training images

Compute arithmetic average (mean face Ψ) of the training image vectors at each pixel point given by:

$$\Psi_{\square \times 1} = \frac{1}{\square}\sum_{\square+1}^{\square}\Gamma i \qquad (13)$$

Obtain mean subtracted vector (Φ) by subtracting the mean face from the each training image vector as given below:

$$\Phi_{i=} \Gamma_I - \Psi \qquad (14)$$

Create the difference matrix (A) which is the matrix of all the mean subtracted vectors and is given by:

$$Ap\times M = \{\Phi_1\Phi_2\ldots\ M\} \qquad (15)$$

Compute the covariance matrix (X) given by:

$$Xp\times p = A.A^T \qquad (16)$$

Compute the eigenvector and eigenvalue of the covariance matrix (X). The dimension of covariance matrix (X) is P × P = (Nx . Ny) × (Nx . Ny). For an image of typical size, the task of computing eigenvector of a matrix of such huge dimension is a computationally intensive task. Instead of using M eigenfaces, M' << M of the eigenfaces can be used for the eigenface projection. This is achieved to eliminate some of the eigenvectors with small eigenvalues, which contribute less variance in the data. Eigenvectors can be considered as the vectors Pointing in the direction of the maximum variance and the value of the variance the eigenvector represents is directly proportional to the value of the eigenvalue (i.e, the larger the eigenvalue indicates the larger variance the eigenvector represents).

Hence, the eigenvectors are sorted with respect to their corresponding eigenvalues. The eigenvector having the largest eigenvalue is marked as the first in the eigenvector matrix. In the next step, the training images are projected into the eigenfaces space and thus the weight of each eigenvector to represent the image in the eigenfaces space is calculated. This weight is simply the dot product of each image with each of the eigenvectors.

Determine the projection of a training image of the eigenvectors as given below:

$$\omega_k = \upsilon^T_k \cdot \Phi = \upsilon^T_k \cdot (\Gamma - \Psi) \qquad k=1,2,\cdots,Mt \qquad (17)$$

Determine the weight matrix -which is the representation of the training images in eigenface space

$$\Omega M'\times 1 = [\omega_1\omega_2\cdots\omega M']^T \qquad (18)$$

The training operation of PCA algorithm ends with the computation of the weight matrix. At this point, the images are just composed of weights in the eigenfaces space, simply like they have pixel values in the image space. The important aspect of the eigenfaces transform lies in this property. Each image is represented by an image of size in (Z_x Z_y) the image space, whereas the same image is represented by a vector of size (M' × 1) in the eigenfaces.

6.2 Recognition operation in PCA

When a new probe(test) image is to be classified, it is also mean subtracted and projected onto the eigenfaces space and the test image is assumed to belong to the nearest class by calculating the Euclidean distance of the test image projections to that of the training image projections.

Let T be an image of size (N_x, N_y,) pixels, then the recognition operation of PCA algorithm can be expressed in mathematical terms as:

Convert the test image matrix *T* of size (N*x*, *Ny)* pixels (*the size of test image must be same as that of the training images)* to the image vector of size (P×1) where *P* = Zx × Zy (i.e: the test image vector is constructed by stacking each column of the test image matrix *T*)
Obtain mean subtracted test image vector (Φ_T) by subtracting the mean face (computed during the training session) from the test image vector as given below:
$\Phi_T\ (P\times 1) = \Gamma_T - \Psi$
Determine the projection of a test image on each of the eigenvectors as given below:

$$\omega_k = \upsilon^T_k \cdot \Phi_T = \upsilon^T_k \cdot (\Gamma_T - \Psi) \qquad k=1,2,\cdots,M' \qquad (19)$$

Determine the weight matrix Ω -which is the representation of the test image in eigenfaces space using (19)
Compute the value of similarity function of given test image for each training image. Similarity of test image with ith training image is defined as:

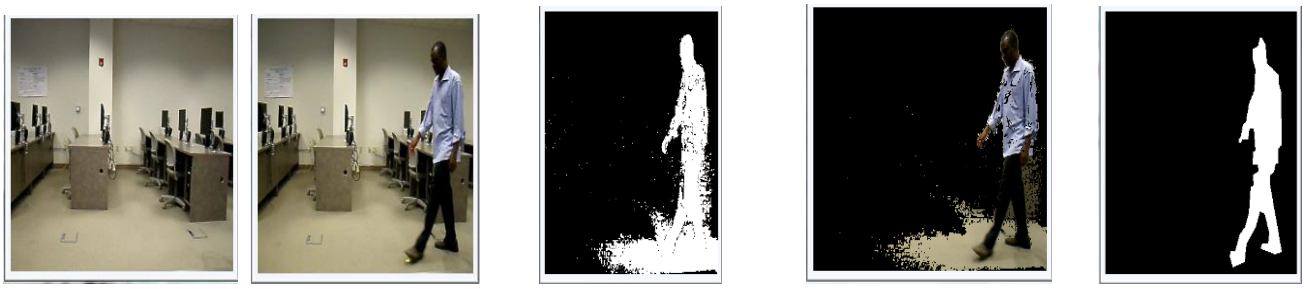
$$\delta_i = \|\Omega_T - \Omega_{\Psi_i}\| = \sqrt{\sum_{k=1}^{M'} (\Omega_T - \Omega_{\Psi_i})^2} \qquad (20)$$

Where (δ_i) is the

Euclidean distance (L2 norm) between projections of images in face space. The training face image which has the minimum distance (L2 norm) is the face that is closely matching with the test face.

5.0 Experiment

A video camera was stationed inside a computer laboratory FUTA to capture the video of students in the classroom for over one hour. Viola and Jones Algorithm with motion was used to detect some frontal faces in the video. Appendix 1.0 shows some of the detected faces. Four images each of fourteen individuals were then used to train an eigenfaces recognition algorithm.



(a) background **(b)Moving objects moved** **(c) Noisy Mask with shadow** **(d) Colored foreground with shadow** **(e)Foreground Mask with shadow removed and morphological operation performed**

Figure 4.0 Motion detection pipeline

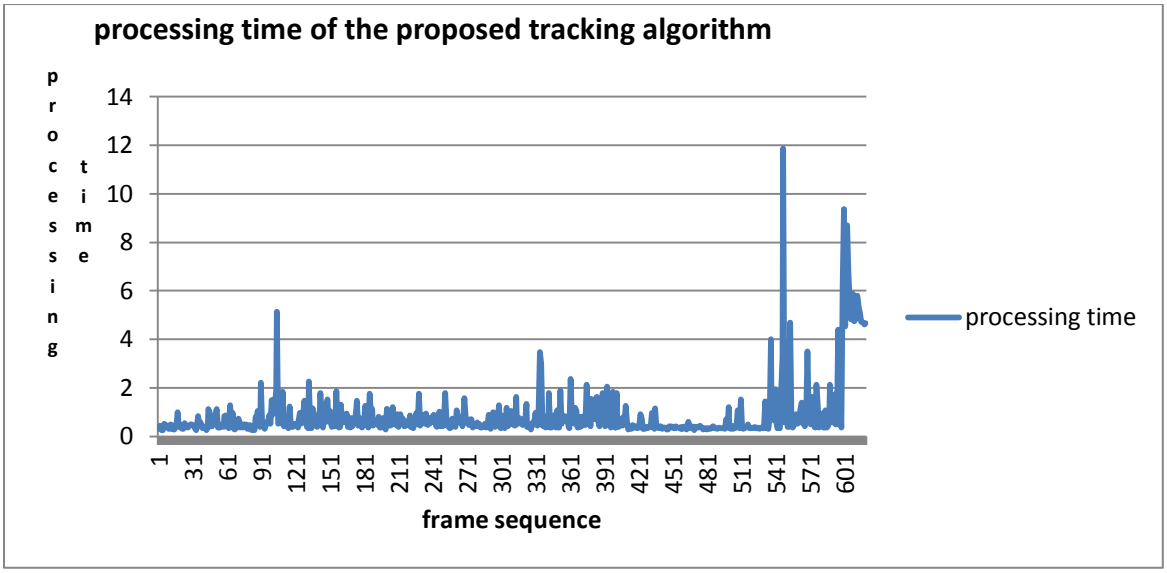


Figure 5.0 Tracking time per frame

Figure 6a Sample True Positive Faces detected from the experiment

As we can see these faces varies in orientation and facial expressions.
To recognize such faces, we perform training and testing separately. Each subject has four faces. We have twelve subjects in the database. Eight were authentic users while four were used for intrusion detection. We trained the algorithm with eight subjects.

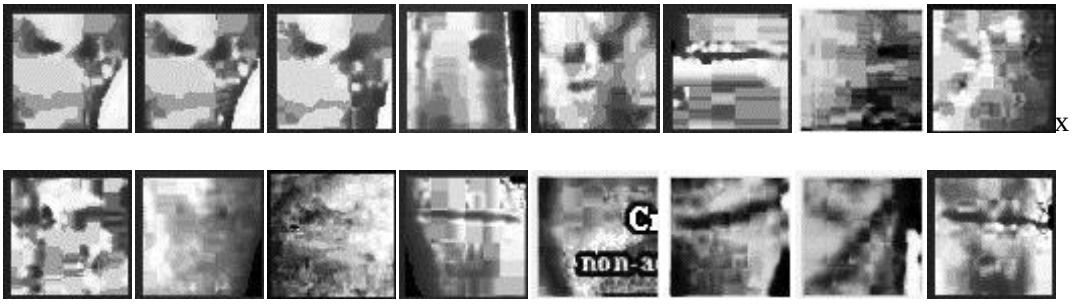


Figure 6b Sample False positive face detected from the experiment

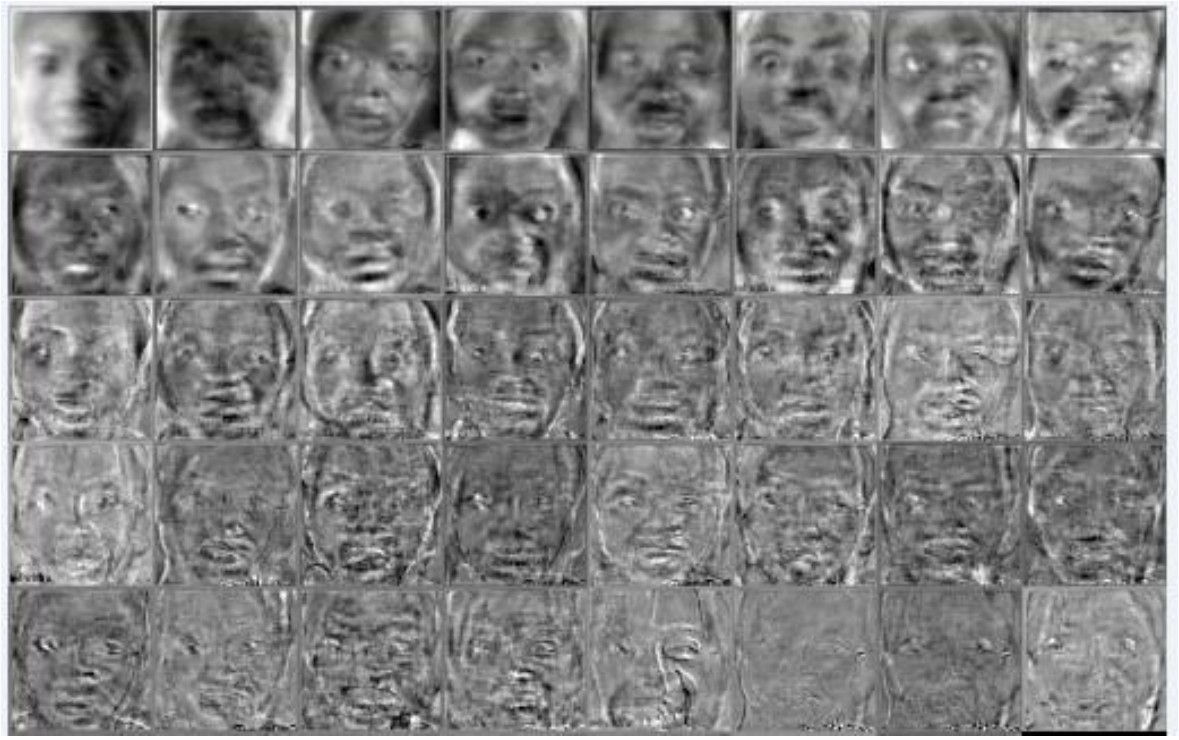


Figure 7.0Some of the Eigenfaces of the training face images

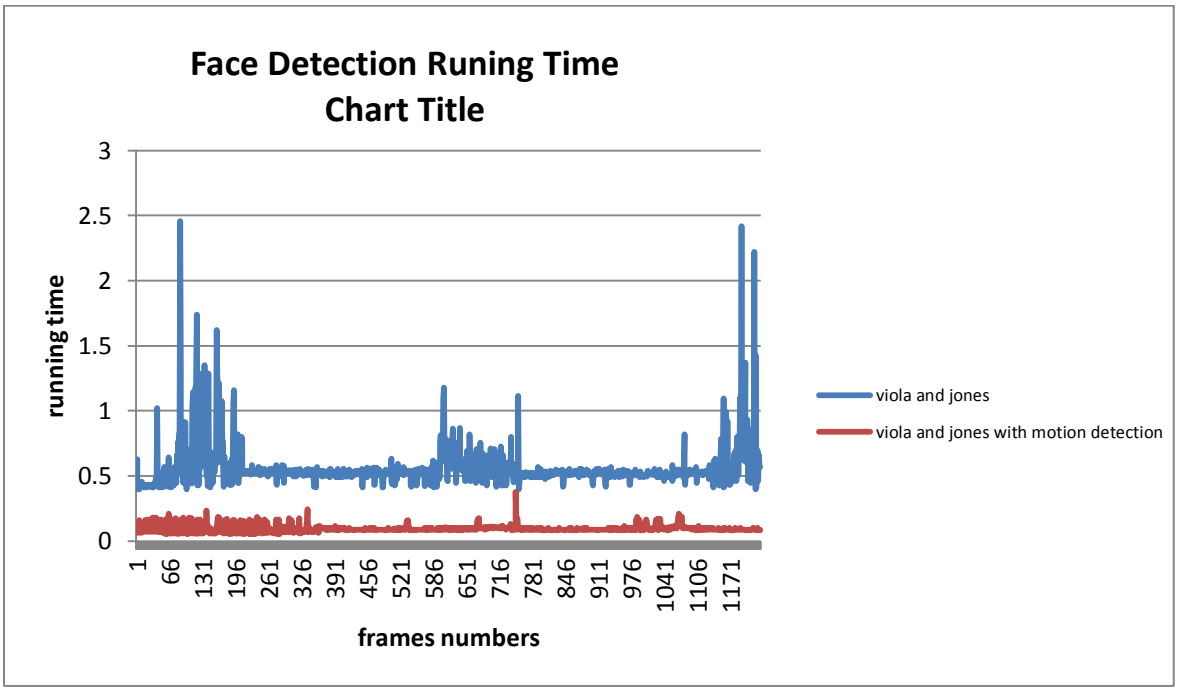


Figure 8.0 processing time of viola and Jones and viola and Jones with motion detection

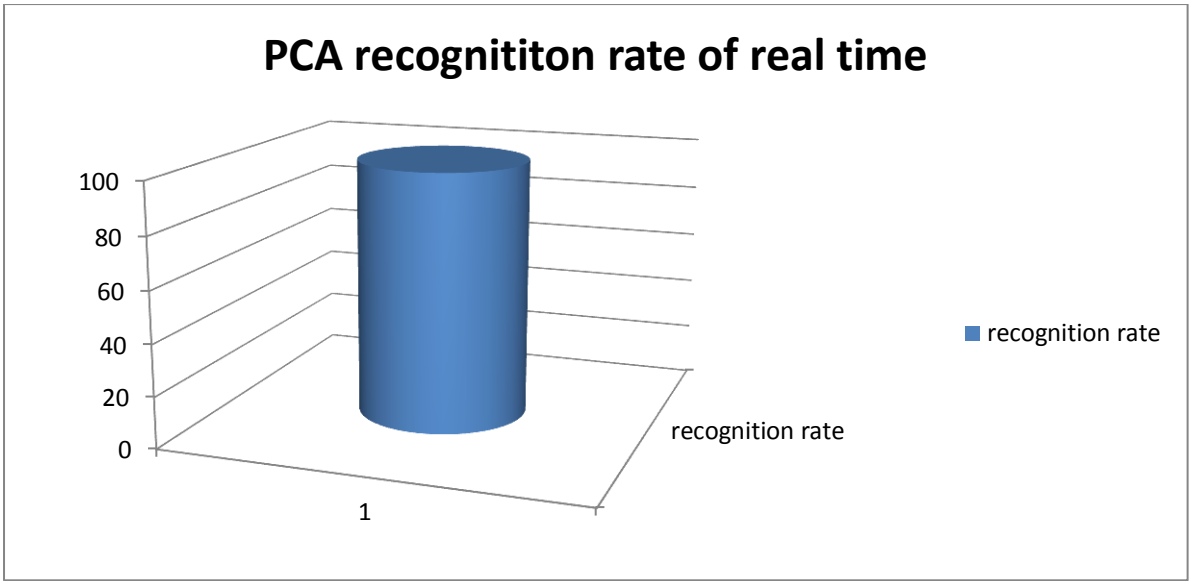


Figure 9.0 Recognition Rate of PCA algorithm (100%)

Figure 10.0 Sample faces detected and recognized by the proposed systems

6.0 CONCLUSION

We have implemented face detection and recognition algorithm using viola and Jones algorithm together with motion detection. It is observed that motion detection before applying viola and Jones algorithms significantly reduced processing time. This will be highly appreciated in a surveillance environment where time is a critical requirement. It can be seeing that principal component analysis works fine in recognizing such faces as we can achieve good accuracy of 100% with the recognition. Future work can be done on this study by using different other classifiers such as neural networks and support vector machines and K Nearest Neighbour. The system has provided an hybridized probabilistic models to enhance fast human recognition using human face as features.

REFERENCES

[1] Anik K. Jain, Arun Ross, Salil Prabhakar (2004) "*An Introduction to Biometric Recognition"* IEEE transactions on circuits & systems for Video Technologies (2004).

[2] Caetano T.S and Barone D.A.C A probabilistic Model for the Human Skin Color ICIAP01, 2001, pp.279-283.N.Oliver, A.

[3] Cox, I., Goshn, J., & Yianilos, P. (1996). Feature-based face recognition using mixture distance. *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 209-216.
[4] Elgammal A., Duraiswami R, Harwood D., and Davis L.S Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance, PROCEEDINGS OF THE IEEE, VOL. 90, NO. 7, JULY 2002.
[5] Kadoury K, Martin D. L. (2006) Face detection in gray scale images Using locally Linear Embeddings. Elsevier Inc. all rights reserved Doi: 10.1016/j. cviu. 2006.06.009. Journal: Computer Vision and Image Outstanding 105 (2007) 1-20
[6] Kanade T. (1973), "Picture Processing System By Computer & Recognition of Human face". A PhD thesis, Department of Information Science, Kyoto University.
[7] Roberto Brunelli and Tomaso Poggio (1998). Face Recognition: Features versus Templates. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(10):1042–1052, October 1998.
[8] Roweis S.T., and Saul (2000) "Non Linear dimensionality reduction by *Locally Linear Embedding*", Science 290: 2323-2326
[9] Rowley, Henry A., Shumeet Baluja and Takeo Kanade (1998). Neural network-based face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 20(1), 23– 38.
[10] Turk, M.A. and A.P. Pentland (1991). Face recognition using eigenfaces. In Proc. of IEEE Conference on Computer Vision and Pattern Recognition pp. 586 – 591.
[11] Viola, Paul and Michael Jones (2001). Rapid object detection using boosted cascade of simple features. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition 2001.