TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

REAL TIME PERSON IDENTIFICATION AND LOCATION DETECTION USING FACE RECOGNITION

Submitted By

Gyanendra Shrestha	**(Exam Roll No: 45462)**
Sanjaya Raut	**(Exam Roll No: 45478)**
Saroj Paudel	**(Exam Roll No: 45479)**
Sudin GC	**(Exam Roll No: 45487)**

THIS REPORT WAS SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULLFILLMENT OF THE REQUIREMENT FOR THE BACHELOR’S DEGREE IN ELECTRONICS & COMMUNICATION ENGINEERING

KATHMANDU, NEPAL

August, 2016

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

STUDENT'S DECLARATION

We hereby declare that we are the only author of this complete work and that no sources other than the listed here have been used in this work.

Name of Students:		**Signature:**
Gyanendra Shrestha	(Exam Roll No: 45462)	___________________
Sanjaya Raut	(Exam Roll No: 45478)	___________________
Saroj Paudel	(Exam Roll No: 45479)	___________________
Sudin G.C.	(Exam Roll No: 45487)	___________________

August, 2016

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

THAPATHALI CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINNERING

SUPERVISOR’S RECOMMENDATION

I hereby recommend this dissertation prepared under my supervision by **Er. Shanta Maharjan** entitled “**Real Time Person Identification And Location Detection Using Face Recognition**” in partial fulfillment of the requirements for the degree of Bachelor in Electronics and Communication Engineering be processed for the evaluation.

____

Er. Shanta Maharjan
August 30, 2016

DEPARTMENTAL ACCEPTANCE

The report entitled **"Real Time Person Identification And Location Detection Using Face Recognition",** submitted by **Gyanendra Shrestha, Sanjaya Raut, Saroj Paudel** and **Sudin G.C.** in partial fulfillment of the requirement for the award of the degree of "Bachelor in Electronics and Communication Engineering" has been accepted as a bona fide record of work independently carried out by them in the department.

Er. Janardan Bhatta

Head of Department

Department of Electronics and Computer Engineering

Thapathali Campus, Tribhuvan University

Kathmandu, Nepal.

CERTIFICATE OF APPROVAL

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering, Institute of Engineering, Thapathali Campus, a final project work entitled "**Real Time Person Identification And Location Detection Using Face Recognition**" submitted by Gyanendra Shrestha, Sanjaya Raut, Saroj Paudel and Sudin G.C. in partial fulfillment of the requirement for Bachelor's Degree in Electronics and Communication Engineering.

____

Er. Janardan Bhatta

Head of Department

Department of Electronics and Computer Engineering, Thapathali Campus

____

External Examiner

Prof. Dr. Shashidhar Ram Joshi

Department of Electronics and Computer Engineering, Pulchowk Campus

____

Project Co-ordinator

Er. Umesh Timalsina

Department of Electronics and Computer Engineering, Thapathali Campus

DATE OF APPROVAL: August, 2016

TABLE OF CONTENTS

ACKNOWLEDGEMENT

Our sincere gratitude goes to Head of Department, Er. Janardan Bhatta at the Department of Electronics and Computer Engineering, Thapathali Campus, IOE for providing us the necessary guidelines for the project. A very special and hearty thanks and gratefulness to our project coordinator Er. Umesh Timalsina for the proper guidance and support for the completion of the project. A wholehearted thanks to our supervisor, Er. Shanta Maharjan for helping us in the project in all ways with support for understanding the feasibility and technical aspects of our project.

A very special thanks goes to Thapathali Campus and Department of Electronics and Computer Engineering for providing us the opportunity to do the project. Without the help of the particulars mentioned above, this project would not have been completed successfully.

ABSTRACT

This project presents an approach to the detection of human facial features and thereby the recognition of the person in real time. The project can be used in security as well as image based query search problems. Our system tracks the subject's face and then recognizes the person and identifies his/her current location by comparing the characteristics of the person's face to the ones available in our database. Faces may be described by a 2-D characteristic view and the face images are turned onto a feature space using the subspace LDA which encodes the distinct facial features and variations. The system contains steady camera located in various locations which is used to capture real time video. The system recognizes the faces in the video and firstly it checks whether the person that is intended to be identified is in video or not. The process continues for all camera. Then if the person to be found out is in within the range of the camera then the location of the person is displayed in the web application which is interfaced in our system.

Keywords: Facial features, Recognition**,** Real time, Subspace LDA, Web application

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2D	Two Dimensions
LDA	Linear Discriminant Analysis
PCA	Principal Component Analysis
KNN	K-Nearest Neighbor
USB	Universal Serial Bus
XML	Extensible Markup Language
OpenCV	Open source Computer Vision
SVM	Support Vector Machine
RAM	Random Access Memory
IP	Internet Protocol
IDE	Integrated Development Environment
FCL	Framework Class Library
BSD	Berkeley Software Distribution
OS	Operating System
SQL	Structured Query Language
IIS	Internet Information Server
MVC	Model View Controller
PGM	Portable Grey Map

1. PROJECT OVERVIEW

1.1. Introduction

In recent technology the popularity and demand of image processing is increasing due to its immense number of application in various fields. Most of these are related to biometric science like face recognitions, fingerprint recognition, iris scan, and speech recognition. Among them face detection and recognition is a very powerful tool for video surveillance, human computer interface, face recognition, and image database management. There are a different number of works on this subject.

Face detection has been regarded as the most complex and challenging problem in the field of computer vision, due to the large intra class variations caused by the changes in facial appearance, lighting and expression. Such variations result in the face distribution to be highly nonlinear and complex in any space which is linear to the original image space. Moreover, in the applications of real life surveillance and biometric, the camera limitations and pose variations make the distribution of human faces in feature space more dispersed and complicated than that of frontal faces. It further complicates the problem of robust face detection .Face detection techniques have been researched for years and much progress has been proposed in literature. Most of the face detection methods focus on detecting frontal faces with good lighting conditions.

Face recognition is a rapidly evolving technology, which has been widely used in forensics such as criminal identification, secured access, and prison security. Face recognition has been one of the most interesting and important research fields in the past two decades. The reasons come from the need of automatic recognitions and surveillance systems, the interest in human visual system on face recognition, and the design of human-computer interface, etc. These researches involve knowledge and researchers from disciplines such as neuroscience, psychology, computer vision, pattern recognition, image processing, and machine learning, etc.

1.2. Motivation

Over the recent years, detecting human beings in a video scene of a surveillance system is attracting more attention due to its wide range of applications in abnormal event detection, human gait characterization, person counting in a dense crowd, person identification, gender classification etc. The machine learning and computer graphic communities are also increasingly involved in face recognition. Besides, there are a large number of commercial, securities, and forensic applications requiring the use of face recognition technologies.

Face recognition has attracted much attention and its researches are rapidly expanded by not only engineers but also neuroscientists, since it has many potential applications in computer vision communication and automatic access control system.

Research in face recognition is motivated not only by the fundamental challenges this recognition problem poses but also by numerous practical applications where human identification is needed. Face recognition, as one of the primary biometric technologies which became more important owing to rapid advances in technologies such as digital cameras, the internet and mobile devices, and increased demands on security. The currently prevailing researches and applications in many fields have motivated us to build an automated system.

1.3. Problem Definition

In this 21st century, a lot of inventions and creations have emerged in the field of science and technology. This emergence is narrowing the world day by day. Now, people can access information around the world while sitting at home. This is just an instance of world moving towards more sophisticated and automated life. While the world is walking with automated technology hand in hand, investigation and security personnel are still relying on traditional technique of checking hours of video footages to guarantee the location of person at real time. Also, searching particular people in an institution, government offices, industries, is a hectic task. Even concerned authority can’t search for details of people working in their own institution. Locating a person, within the premises, in such large institutions come at expense of hours of searching. So, in order to solve these difficulties, this system is build.

1.4. Objectives

The main objectives of this project are:

- To develop a system that is capable of real time face detection and recognition.
- To create an image based searching system for searching and locating person.
- To build a web application based searching of person.

1.5. Scope

The scope of this project are listed below as:

- To build a system that solves the need of manpower for checking hours of video footage in surveillance and security systems.
- This project can be helpful for real time searching of a person and his/her location.
- This system can be used in searching the criminals, lost persons etc.

2. LITERATURE REVIEW

2.1. Biometric Authentication

Biometrics is an emerging field which in recent times is mostly researched for person authentication. There are various biometric systems as iris, signature, fingerprint, retina and face. [1] provides basic overview of biometric system and its types that can be employed for authentication. Researchers have been busy in their researches to identify which one among various approaches is best suited for particular problems. So, [1] emphasizes that best biometric characteristics is determined in terms of robustness, distinctiveness, acceptability, accessibility and availability. [2] has made a comparison of various biometrics approaches on basis of characteristics of [1], and has concluded that iris and face serve as best method for authentication. It further elaborates that face recognition method of authentication being hands-free and non-intrusive along with its high performance and acceptability, is best suited for authentication.

2.2. Face Recognition

Computer vision enthusiasts and researchers have been involved in face recognition problem since a long time. So, many systems have been developed for addressing the problem. [3] guides us through the steps involved in solving face recognition problem. It partitions the face recognition problem mainly into face detection, feature extraction and classification.

2.2.1. Face Detection

First step of face recognition system is face detection which is simply to check whether the input still image or a video frame consists of a face or not. It also involves extracting face portions only from an image. [4] provides an insight into various measures that can be applied to face detection problem. It also gives an overview of these measures and conceptual idea for choosing suited method as per system requirement. [5] provides basic entities of Voila Jones algorithm and its working mechanism. [6] uses Voila-Jones algorithm for rapid face detection method based on a boosted cascade of simple classifier. These methods use Haar-like features [7] and Adaboost algorithm [8] to achieve fast and stable face detection. Haar-like feature is one of the powerful tools for face detection. It depicts that Voila Jones use Haar Cascade

Classifiers along with Adaboost algorithm. The author of [6] creates a new framework that is demonstrated to the task of face detection. It covers mainly three parts.
They are fast feature evaluation, AdaBoosting by selecting only critical features and combining complex classifier in a cascade structure.

2.2.2. Feature Extraction

It is the process of extracting features from face images for the classification of faces which later results in face recognition. Different algorithms have been developed for this purpose as PCA, LDA and so on. [9] performs a comparison on face recognition algorithms and concludes LDA (with an accuracy of 91.7%) is better than PCA (with an accuracy of 80.75%) in neutral schema. [10] uses LDA for feature extraction and provides a general overview of the algorithm for using it in a face recognition problem. Authors of [11] classify the process into three parts: creating a large group of subjects with diverse facial characteristics with minor variation in view angle containing face region only, vector expansion and creating framework that performs a cluster separation analysis in the feature space so that it calculates the scatter matrices within and between the classes. [12] provides a way to deal with the so-called small sample size problem of LDA [13]. This problem arises whenever the number of samples is smaller than the dimensionality of the samples. Under these circumstances, the sample scatter matrix may become singular, and the execution of LDA may encounter computational difficulty.

2.2.3. Classification of Features

Classification of features refers to the process of identifying the class of recognition problem which on assignment yields face recognition. It can also be done by various methods. Two of the popular machine learning approaches are SVM and KNN. Both are equally good and anyone out of two can be applied. K-nearest neighbors (KNN) rule is one of the oldest and simplest methods for pattern classification. Author of [14] emphasizes that KNN rule classifies each unlabeled example by the majority label among its k-nearest neighbors in the training set. Its performance thus depends crucially on the distance metric used to identify nearest neighbors According to author [14], k-nearest neighbors always belong to the same class while examples from different classes are separated by a large margin.

3. METHODOLOGY

3.1 Theoretical Background

3.1.1. Face Extraction using AdaBoost and Cascaded Detector

Viola and Jones were first to present a state-of-the-art performance, real-time face detection system. This concepts allow for highly accurate object and face detection at high speeds. Haar wavelets form the basic features for face detection, they are simple but fast to compute. There are three key contributions. The first is the introduction of integral image for rapid computation of the features used in the detector. The second is the AdaBoost algorithm for selection of efficient classifiers from a large population of potential classifiers. Third is the method for combining the classifiers generated by AdaBoost algorithm into a cascade, which has the property of removing most of the non-face images in the early stage by simple processing, and focus on complex face like regions in the later stages which take higher processing time. This algorithm is selected for the face detection because of its property of detecting faces extremely rapidly which would help in our project in processing real time videos.

3.1.2. Haar Features Selection

Haar features are simple rectangular features reminiscent of Haar basis functions first suggested for use in face detection by Papageogiou et al. in [18]. The over-complete feature set consists of weighted combinations of two or more rectangles' intensity sums. Individual features can described as

$$feature_I = \sum_{i \in I = \{1,\ldots..,N\}} w_i s(r_i) \qquad (3.1)$$

with weights w_i, rectangles r_i and number of rectangles N, the rectangle area intensity sum function $s(r_i)$. The weights are restricted to opposite signs in order to yield rectangle area differences, the number of rectangles is usually restricted to a maximum of four, although more elaborate features are possible. The features resemble Haar-basis functions, they are constructed to capture edges, lines and diagonals, sometimes center-surround features are added to the set.

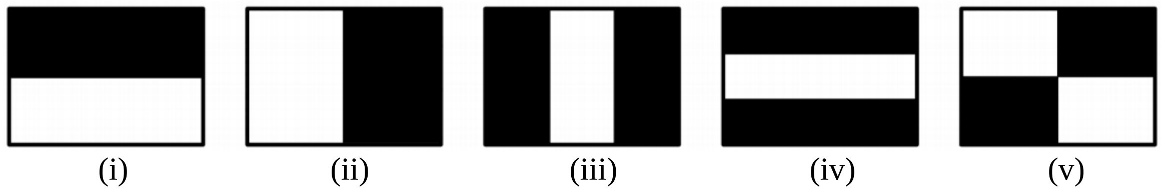


Figure 3.1: Features used in AdaBoost algorithm [19]. (i) and (ii) are two rectangle features, (i) being two vertically stacked rectangular feature. Similarly, (iii) and (iv) are three rectangle feature and (v) is the four rectangle feature.

Given the base resolution of the detector sub-window is 24x24, the exhaustive set of rectangle features is quite large, more than 160,000. In simple words, the features have to be generated such that it starts from every possible pixel of the detector sub-window and also should cover all the widths and heights possible. This should be done for all types of features (two rectangle, three rectangle and four rectangle features). The algorithm for the feature generation is given below:

1. Start from the initial position of the sub-window, (1, 1). Create a rectangular feature of size such that 1 pixel represents the black region and 1 pixel represents the white region.
2. Move the feature throughout the sub-window.
3. Increase the size of the feature such that both the black region and white region size increases by 1 in horizontal direction. Go to step 2, until the width of the feature equals the width of the sub-window
4. Increase the size of the feature such that both the black region and dark region size increases by 1 in vertical direction. Go to step 2 until feature height equals the height of the detector sub-window
5. For all the starting position and all the width and height of the feature (from steps 1 to 4), store the co-ordinates as (x, y, w, h) where x= starting x-coordinate, y = starting y-coordinate, w = width and h = height of the feature
6. Repeat steps 1 to 5 for all types of rectangular features (i.e. two rectangle, three rectangle and four rectangle feature).

3.1.3. Creating an Integral Image

The input image is converted into integral form. This is done by making each pixel equal to the entire sum of all pixels above and to the left of the concerned pixels.

1	1	1
1	1	1
1	1	1

(i)

1	2	3
2	4	6
3	6	9

(ii)

Figure 3.2: (i) Input Image and (ii) Integral image

This allows for the calculation of the sum of all pixels inside any given rectangle using only four values. These values are the pixels in the integral image that coincide with the corners of the rectangle in the input image.

Integral image size is the same size of the image, but the value at any location (x, y) contains the sum of the pixels above and to the left of x, y, inclusive as shown in the figure.

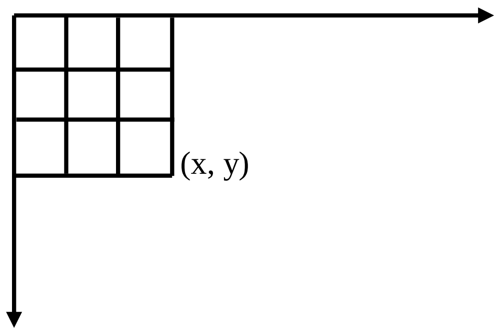


Figure3.3: Integral image is the sum of the pixels at the left and above of the point (x, y) inclusive.

$$ii(x,y) = \sum_{x' \le x y' \le y} \mathrm{i}(\mathrm{x}',\mathrm{y}') \qquad (3.2\ )$$

where, $ii(x, y)$ is the integral image and $i(x, y)$ is the original image. So, for the computation of $ii(x, y)$, following pair of recurrences can be used:

$$s(x, y) = s(x, y - 1) + i(x, y) \tag{3.3}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \tag{3.4}$$

Where $s(x, y)$ is the cumulative row sum and $s(x, -1) = 0$, and $ii(-1, y) = 0$.

3.1.4. AdaBoost Training

In a single sub window of size 24×24, there are more than 160,000 features. Computation of all these features in detection can be very expensive in time. But there are efficient classifiers among these large number of features which when efficiently combined gives a robust classifier. The selection and combination of these efficient classifiers from among the large set is done by AdaBoost algorithm. Viola and Jones have given a variant of AdaBoost algorithm to train the classifiers. AdaBoost algorithm is given a feature set with large number of features and the training images (labeled faces and non-faces, which is supervised learning). AdaBoost learning algorithm is used to boost the classification performance of simple learning algorithm (e.g. a simple perceptron). It does this by combining a collection of weak classification functions to form a strong classifier. In the language of boosting the weak classification functions are called weak learners, because we do not expect even the best weak learner to classify the training data well (that is, for a given problem even the best perceptron may classify the training data correctly only 51% of time). The selection of the best classifier for each round is the one which gives lowest classification error in the training data. Boosting is the method of combining the weak classifiers to form a strong classifier. In the later round of learning, AdaBoost finds the new weak classifier after re-weighting the training examples such that it emphasizes on the examples incorrectly classified by the previous weak classifiers. The weak classification function selects a single rectangular feature which best separates the positive and negative examples, determines the threshold and gives a weak classifier. So, weak classifier ($h_j(x)$) consists of a rectangular feature (f_j), threshold (θ_j) and a polarity (p_j) indicating the direction of inequality.

$$h_j(x) = \begin{cases} 1, if\ p_j . f_j(x) < p_j . \theta_j \\ 0\ otherwise \end{cases} \quad (3.5)$$

Boosting algorithm which when run for T number of rounds selects T number of best weak classifiers and combines these to form a strong classifier also finding the threshold for the strong classifier. The boosting algorithm used for the selection of the rectangular 23 feature and combining those to form a strong classifier is given below:

1. Given example images $(x_1,y_1), \ldots, (x_n, y_n)$ where $y_i = 0,1$ for negative and positive examples respectively
2. Initialize weights $W_{1,i}=1/2m,\ 1/2l$ for $y_i = 0,1$ respectively, where m and l are the numbers of negatives and positives respectively
3. For t= 1, . . . , T :
   a) Normalize the weights

      $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$ so that w_t is a probability distribution

   b) For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t

      $\in_j = \sum_i w_i |h_j(x_i) - y_i|$

   c) Choose the classifier, h_t, with the lowest error $\in_t$
   d) Update the weights:

      $w_{t+1,i} = w_{t,i} \beta_t^{1-\in_i}$
4. The final strong classifier is:

$$h(x) = \begin{cases} 1, if\ \sum_{t=1}^{T} \alpha_t h_t(x) \geq\ 0.5 \sum_{t=1}^{T} \alpha_t \\ 0\ otherwise \end{cases} \quad (3.6)$$

where $\alpha_t = log \frac{1}{\beta_t}$

3.1.5. Cascading Classifier

The basic principle of the Viola-Jones face detection algorithm is to scan the detector many times through the same image – each time with a new size. Even if an image should contain one or more faces it is obvious that an excessive large amount of the evaluated sub-windows would still be negatives (non-faces). This realization leads to a different formulation of the problem. Instead of finding faces, the algorithm should discard non-faces. The thought behind this statement is that it is faster to discard a non-face than to find a face. With this in mind a detector consisting of only one (strong) classifier suddenly seems inefficient since the evaluation time is constant no matter the input. Hence the need for a cascaded classifier arises. The cascaded classifier is composed of stages each containing a strong classifier. The job of each stage is to determine whether a given sub-window is definitely not a face or maybe a face. When a sub-window is classified to be a non-face by a given stage it is immediately discarded. Conversely a sub-window classified as a maybe-face is passed on to the next stage in the cascade. It follows that the more stages a given sub-window passes, the higher the chance the sub-window actually contains a face.

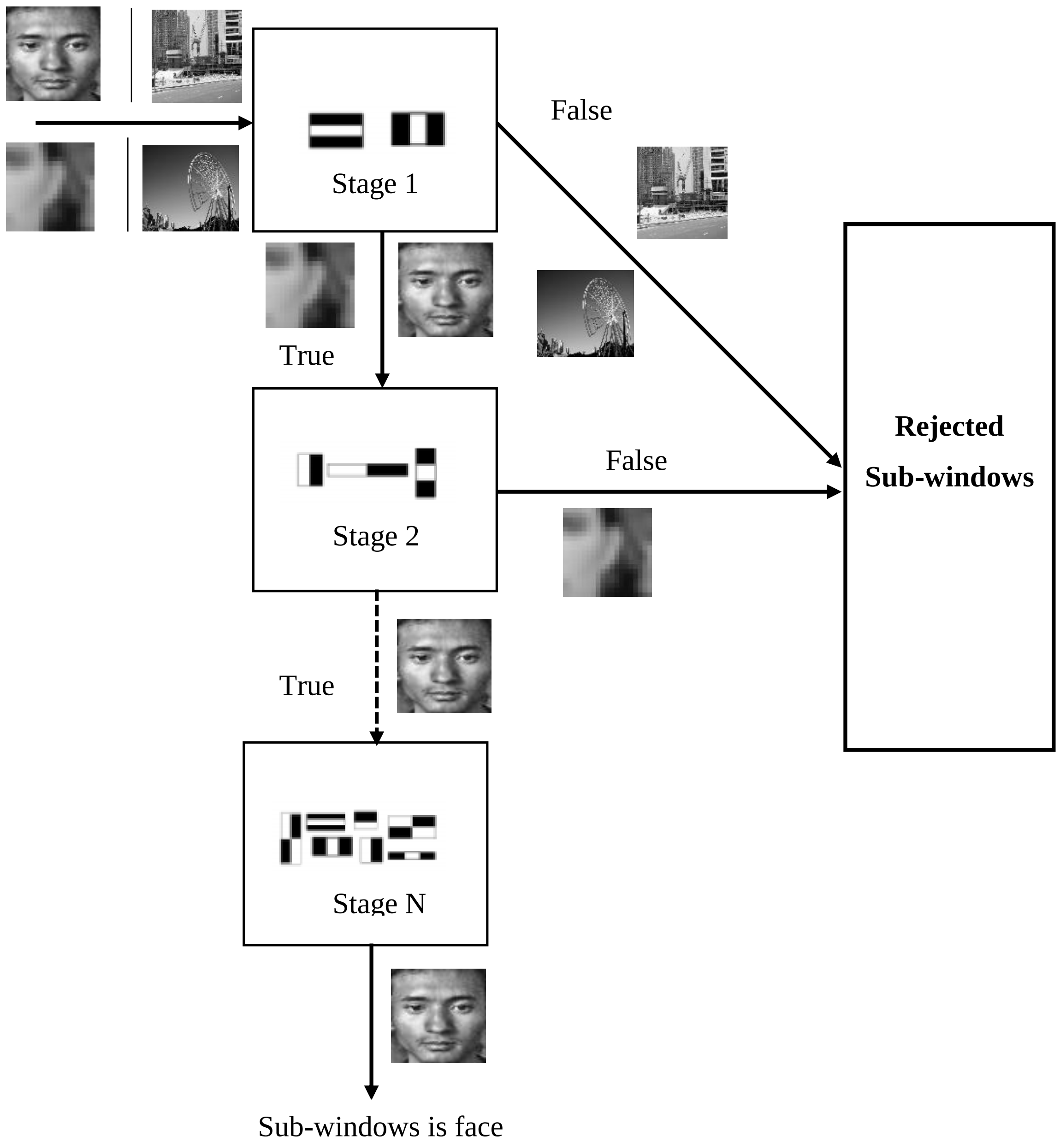


Figure 3.4: Cascade of Classifier for Face Detection

3.1.6. Training the Cascade Classifier

For the training of the cascade of classifiers, each stage 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 larger number of classifiers than the previous. The final false positive rate and the detection rate of the cascade is given as:

$$F = \Pi^{K}_{i=1} f_i$$

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:

$$D = \Pi^{K}_{i=1} d_i$$

Where D is the detection rate of the cascaded classifier, d_i is the detection rate of ith stage and K is the number of stages. Every face detection system requires high detection rate and a low false positive rate. Given the concrete goal for overall false positive and detection rates, target rates can be determined for each stage in the cascade process. For example, detection rate of 0.9 can be achieved by 10 stages with the detection rate of each stage $d_i = 0.99(0.99^{10} = 0.9)$. The detection rate of 0.99 for each stage can be found for a high false positive rate. But for 10 stages, if the false detection rate of each stage is 0.4, then the final false detection rate will be $0.000105(0.4^{10})$.

For the training of the cascade, initially faces and non-faces for the training and faces and non-faces for the validation are provided. AdaBoost algorithm is designed to find a strong classifier which best classifies the faces from the non-faces and is not designed to produce high detection rate with high false detection rate. So, to achieve high detection rate of say 0.99, the stage has to be checked with the validation image to find the current detection rate, and the threshold of the strong classifier has to be reduced to achieve the detection rate of 0.99. In doing so, the false positive rate also increases, and if it is greater than the desired false positive rate (say 0.4), then the stage has to be trained again with more number of features included.

After the generation of one stage, the non-faces training images for the next stage are generated by running the detector up to the previous stage on a set on non-faces containing images. These false detections are then used as non-faces training images for the training of the new stage. This way, the complexity of the non-faces increases as the number of stages increases, resulting in more and more faces-like non-faces training images in later stages. So, the number of classifiers in the stage also increases as the stage number increases.

3.1.7. Training algorithm for cascade detector

The training algorithm is as follows:

```
a) User selects values of f , the maximum acceptable false positive rate per layer and d,
   the minimum acceptable detection rate per layer
b) User selects target overall false positive rate, $F_{target}$.
c) P = set of positive examples
d) N = set of negative examples
e) $F_0 = 1.0; D_0 = 1.0$
f) $i = 0$
g) while $F_i > F_{target}$
   • $i \leftarrow i + 1$
   • $n_i = 0; F_i = F_{i-1}$
   • while $Fi > f \times Fi-1$
       - $n_i \leftarrow n_i + 1$
       - 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
         decrease threshold for the $i^{th}$ classifier until the current cascade classifier
         has a detection rate of at least $d \times D_{i-1}$ this also affects $F_i$)
   • $N \leftarrow \emptyset$
   • 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
```

The algorithm for the generation of cascade of classifiers is shown in the figure above. First, the maximum acceptable false positive rate per stage f, minimum acceptable detection rate per layer d and overall target false positive rate of the final cascade F_{target} has to be defined. Also, the training set of positive P and negative examples N, set of validation positive and negative examples, and the non-face source images from which the non-faces for the later stages of the cascade will be generated, has to be provided. Then the stages of the cascade are generated using AdaBoost algorithm. Once the stage is generated, its threshold is reduced to meet the requirement of minimum detection rate. If reduction of threshold in doing so does not meet the requirement for the maximum false positive rate, more classifiers are added to the stage using AdaBoost algorithm. After these conditions are met, all the non-face training images are removed and new ones are generated from the non-face source images using the cascade generated so far. Then the new stage generation starts. This continues until the final false positive rate is less than or equal to the target false positive rate.

3.1.8. PCA Algorithm

It is a holistic approach based face recognition algorithm. It is also known as Eigenfaces. Any image of dimension $m \times n$ can be thought of as a point in a P dimensional image space having values in the range of pixel values. For the case of gray scale images, in each dimension the image could have a value in between 0 and 255. An image can be thought as a point in the image space by converting the image to a long vector by concatenating each column of the image one after the other. The Eigenface method tries to find a lower dimensional space for the representation of the face images by eliminating the variance due to non-face images; that is, it tries to focus on the variation just coming out of the variation between the face images. Eigenface method is the implementation of Principal Component Analysis (PCA) over 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.

Let $\{X = x_1, x_2, \dots., x_n\}$ be the matrix containing face images. Each image matrix converted into a vector. For example, a $m \times n$ matrix is converted into a vector with $m \times n$ rows. The subspace "eigenfaces" could be obtained by following the below steps:

1. Compute the mean μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3.7}$$

2. Compute the Covariance Matrix S

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \qquad (3.8)$$

3. Compute the eigenvalues λ_i and eigenvectors v_i of S

$$Sv_i = \lambda_i v_i , i = 1,2, \dots \dots, n \qquad (3.9)$$

4. Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues.

   The k principal components of the observed vector x are then given by:

$$y = W^T(x - \mu) \qquad (3.10)$$

   Where W = $(v_1, v_2, \dots \dots \dots \dots \dots \dots, v_n)$. The reconstruction from the PCA basis is given by:

$$x = Wy + \mu \qquad (3.11)$$

3.1.9. LDA Algorithm

LDA is a holistic approach based face recognition method proposed by Etemad and Chellapa [3]. However unlike Eigenfaces which attempts to maximize the scatter of the training images in face space, it attempts to maximize the between class scatter, while minimizing the within class scatter. In other words, moves images of the same face closer together, while moving images of different faces further apart. Overall it tries to increase the ratio of the between class scatter to within class scatter.

LDA is a supervised learning algorithm meaning that the class labels of training set will be used in the training process. Let X be the matrix containing training face images and X_i be the matrix containing face images belonging to class i.

$$X = \{X_1, X_2, \dots \dots \dots ., X_c\}$$

$$X_i = \{x_1, x_2, \dots \dots \dots .., x_n\}$$

The scatter matrices S_B and S_W are calculated as:

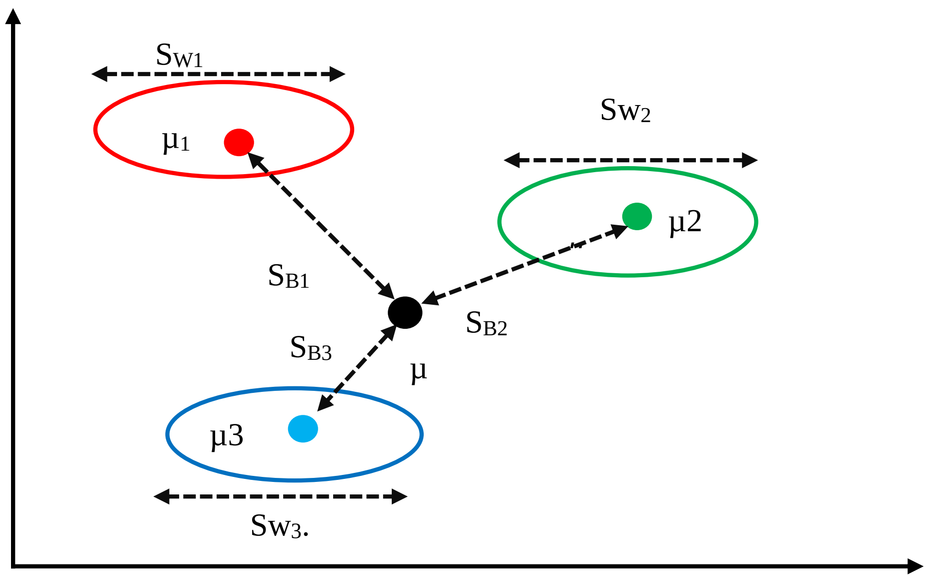


Figure 3.5: Scatter matrices S_B *and* S_W for a three class problem.

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)\,(\mu_i - \mu)^T \qquad (3.12)$$

$$S_W = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_j - \mu_i)^T \qquad (3.13)$$

Where μ is total mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (3.14)$$

And μ_i is the mean of class i ∈ $\{1, \dots \dots .., c\}$:

$$\mu_i = \frac{1}{|X_i|}\sum_{x_j \in X_i} x_j \qquad (3.15)$$

Fisher's algorithm then looks for a projection *W*, that maximizes the class separability criterion:

$$W_{opt} = \arg max_W \frac{|W^T S_B W|}{|W^T S_W W|} \quad (3.16)$$

A solution for this optimization problem is given by solving the general Eigenvalue problem:

$$S_B v_i = \lambda_i S_w v_i$$
$$S_W^{-1} S_B v_i = \lambda_i v_i \quad (3.17)$$

The rank of S_W is at most ($N-c$), with N samples and c classes. In pattern recognition problems the number of samples N is almost always smaller than the dimension of the input data (the number of pixels), so the scatter matrix S_W becomes singular. This was solved by performing a Principal Component Analysis on the data and projecting the samples into the ($N - c$)-dimensional space. A Linear Discriminant Analysis was then performed on the reduced data, because isn't singular anymore.
The optimization problem can be rewritten as:

$$W_{pca} = \arg max_W |W^T S_T W| \quad (3.18)$$

$$W_{fld} = \arg max_W \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_W W_{pca} W|} \quad (3.19)$$

The transformation matrix W, that projects a sample into the (c-1) dimensional space is then given by:

$$W = W_{fld}^T W_{pca}^T \quad (3.20)$$

3.1.10. Subspace LDA

Subspace LDA is a hybrid algorithm. It uses both PCA and LDA. However only LDA is used as the ultimate classifier; PCA is only used as a dimension reduction step. So in subspace LDA algorithm we initially created a PCA subspace using the training images as described above in the PCA algorithm. All the training images are then projected into the PCA subspace. These projections are then input to the LDA which again creates a new subspace from these PCA projections as described above in the LDA algorithm. The PCA projections are again projected into the LDA subspace.

3.1.11. KNN Classifier

The simplest classification scheme is a nearest neighbor classification in the image space. Under this scheme an image in the test set is recognized by assigning to it the label of the closest point in the learning set, where distance are measured in image space. If all images have been normalized to be zero mean and have unit variance, then this procedure is equivalent to choosing the image in learning set that best correlates with the test image. Because of normalization process, the result is independent of light source intensity and the effects of a video camera's automatic gain control. Feature selection is achieved using this learning algorithm by constraining each classifier to depend on only a single feature [14].The Euclidean distance metric [4] is often chosen to determine the closeness between the data points in KNN. A distance is assigned between all pixels in a dataset. Distance is defined as the Euclidean distance between two pixels. The Euclidean metric is the function d: R_n X $R_n \rightarrow R$ that assigns to any two vectors in Euclidean n-space:

$$X= (x_1,\ldots,x_n)$$
$$Y= (y_1,\ldots,y_n)$$

The distance,

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + \cdots \ldots \ldots . + (x_n - y_n)^2} \qquad (3.21)$$

This gives the "standard" distance between any two vectors in R_n. From these distances, a distance matrix is constructed between all possible pairings of points (x, y).

3.1.11.1. KNN Algorithm

1) Each data pixel value within the data set has a class label in the set, Class = $\{c_1,\ldots,c_n\}$.
2) The data points, k-closest neighbors (k being the number of neighbors) are then found by analyzing the distance matrix.
3) The k-closest data points are then analyzed to determine which class label is the most common among the set.
4) The most common class label list is the n assigned to the data point being analyzed.

3.1.11.2. KNN Performance Vs. choice of K

1) When noise is present in the locality of the query instance, the noisy instance(s) win the majority vote, resulting in the incorrect class being predicted. A larger k could solve this problem.

2) When the region defining the class, or fragment of the class, is so small that instances belonging to the class that surrounds the fragment win the majority vote. A smaller k could solve this problem. The KNN shows superior performance for smaller values of k compared to larger values of k.

Instances can be considered as points within an n-dimensional instance space where each of the n-dimensions corresponds to one of the n-features that are used to describe an instance. The absolute position of the instances within this space is not as significant as the relative distance between instances. This relative distance is determined by using a distance metric. Ideally, the distance metric must minimize the distance between two similarly classified instances, while maximizing the distance between instances of different classes. KNN predictions are based on the intuitive assumption that objects close in distance are potentially similar, it makes good sense to discriminate between the k nearest neighbors when making predictions, i.e., let the closest points among the k nearest neighbors have more say in affecting the outcome of the query point. This procedure has several well-known disadvantages. First, if the image in the learning set and test set are gathered under varying lighting conditions, then the corresponding points in the image space will not be tightly clustered. So in order for this method to work reliably under variations in lighting, a learning set which densely sampled the continuum of possible lighting conditions, is required. Second, correlation is computationally expensive. For recognition, we must correlate the image of the test face with each image in the learning set to reduce computational time. Third, it requires large amounts of storage: i.e. the learning set must contain numerous images of each person.

3.2. Tools and Technologies

3.2.1. Tools

- Pycharm IDE: Pycharm is a Python based IDE.
- Visual Studio: Microsoft Visual Studio is an IDE from Microsoft. Our use of Visual Studio is for the development of website for user interface.

3.2.2. Technologies and Platform

- Python 2.7

  Python is widely used high-level, general purpose, interpreted, dynamic programming language. Python supports multiple programming paradigm including object-oriented, imperative and functional or procedural programming styles. We have used Python 2.7 as it has slightly more library support and is easier to code on.
- .NET Framework

  .NET Framework is a framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library known as Framework Class Library (FCL) and provides language interoperability i.e. each language can use code written in other languages across several programming languages. FCL provides programmers produce software by combining their own source code with .NET Framework and other libraries.
- OpenCV

  OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications.
- Windows

  Windows is a personal computer operating system developed by Microsoft, which is the platform our project is implemented in.

4. SYSTEM DESIGN, DEVELOPMENT AND ANALYSIS

4.1. Requirement Analysis

4.1.1. Functional Requirements

The functional requirements of the system are as follows:

- The system should be able to take training data, face images to train the system.
- From the supplied valid training data, system should be able to generate a proper model.
- The system should be able to classify the input test image in satisfactory class.
- The system should be able to provide satisfiable result.

4.1.2. Non-functional Requirements

The non-functional requirements of the system are encompassed here:

- **Performance**

  Since we have used python for scripting, list processing and list array manipulation becomes extremely easy and fast. Also, availability of numpy and scipy kits for python makes vector computation available and hence array and matrix processing can be carried out faster.
- **Accuracy**

  Compared to other methods of model generation and testing, KNN is found to be relatively more reliable and accurate. And our implementation also added a point to this method over other methods.
- **Reliability**

  The system should able to perform with high reliability for change in environmental factors such as various lighting conditions and facial appearance.

4.2 System Flowchart

A user-friendly website has been built which is easily accessible through the end devices such as PCs. At first the query is given by user to system to search the location of desired person which is already in system database. The system takes input video feed from cameras placed at different locations. The data obtained from the image processing module which consist of series face detection and recognition operation is uploaded and stored in the server and final result about the location information of person is provided to end user.

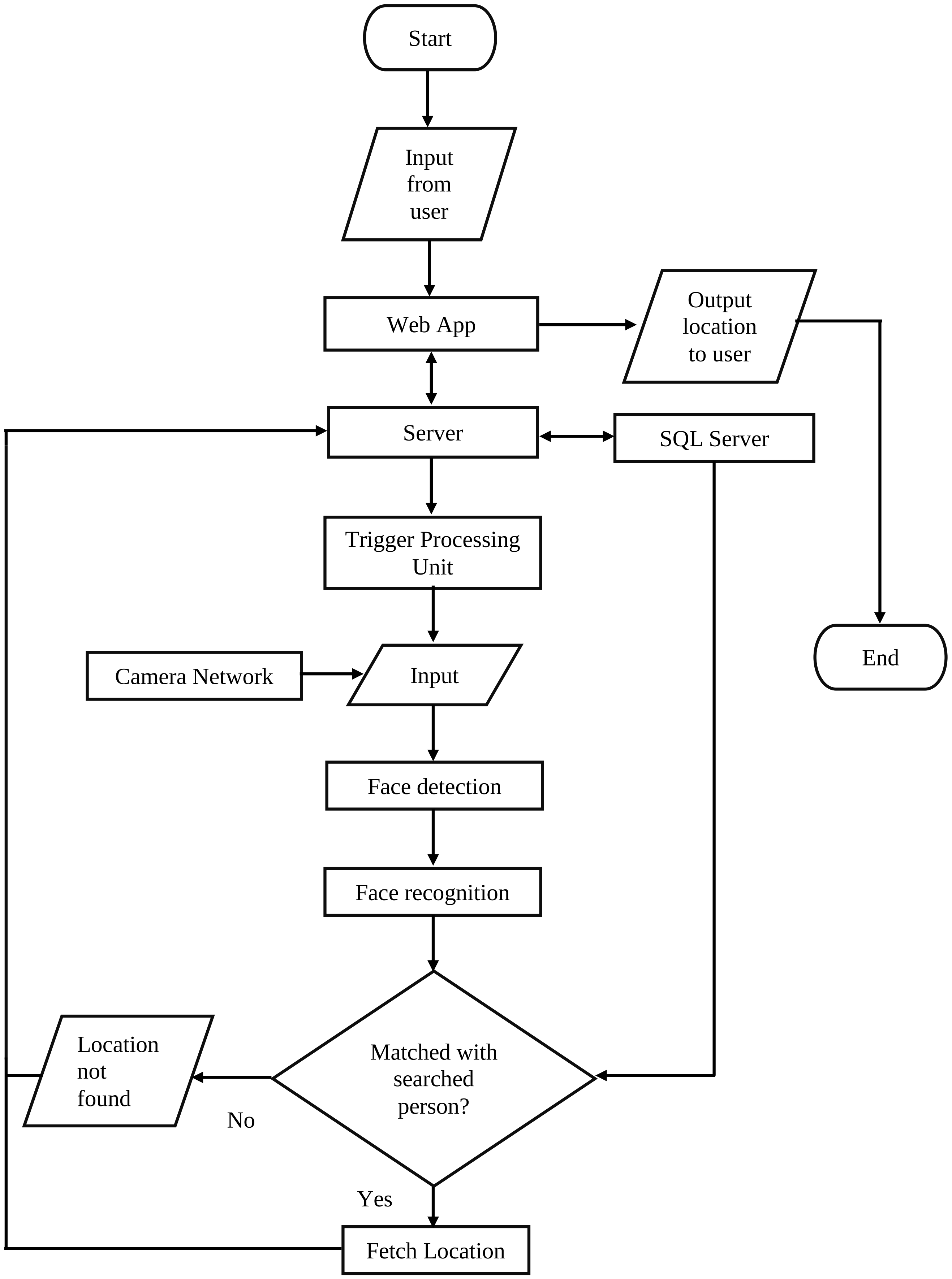


Figure 4.1: System Flowchart

4.3 Implementation

4.3.1. Data Collection

For the purpose of creating haar-like cascade classifier for face detection, 5000 of the faces images (positive images) and 8000 non-face images (negative images) were used initially. Negative images were downloaded from preexisting image database through the internet. The positive and the negative images were resized to 50x50 pixels and 100x100 pixels respectively. About 20,000 positive samples were derived from the original 5000 positive images using the *opencv_createsamples* function of the OpenCV library. Using the *opencv_createsamples* function, the 50x50 sized positive images were rotated and placed on top of 100x100 sized negative samples to create the 100x100 sized positive samples.

We have used ORL face datasets for initial training and testing phase and we created own IOE Thapathali face datasets for final implementation of our system. The ORL data set is made up of ten different images of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). The data set was subdivided into a training set and a test set.

4.3.2. Haar Cascade Training

Training was performed on our 8GB RAM laptop as a server using the collected images. Images were then uploaded on the server (laptop) and training was performed using the opencv*_traincascade* function of the OpenCV library. The number of training stages was set to ten so as to not over-train or under-train the Haar cascade. The under-trained Haar cascade would not detect the desired object of interest whereas the over-trained Haar cascade would falsely categorize the negative samples as object of interest. An .xml file was generated after the completion of the training process. The entire training process took around two days and four hours. The final stage parameters obtained are:

- Acceptance ratio

  The acceptance ratio obtained was 8000:0.0599215. The cascade was not trained further than ten stages as it would result in the decrement of the acceptance ratio resulting in the overtraining of the cascade.
- Number of features

  In the final stage, 74 different features are used to separate the positive samples from the negative samples.
- Hit rate

  Hit rate of 0.9953 was obtained in the final stage. Hit rate of 99.53% indicates that 99.53% of the positive samples are successfully detected as the object of interest.
- False alarm rate

  The false alarm rate obtained in the final stage was 0.483, which indicates that 48.3% of negative samples are falsely categorized as positive samples.

4.3.3. Face Detection

After Haar Training, an .xml file was obtained from the training process. This xml file is used to create a cascade classifier object for the detection of face.

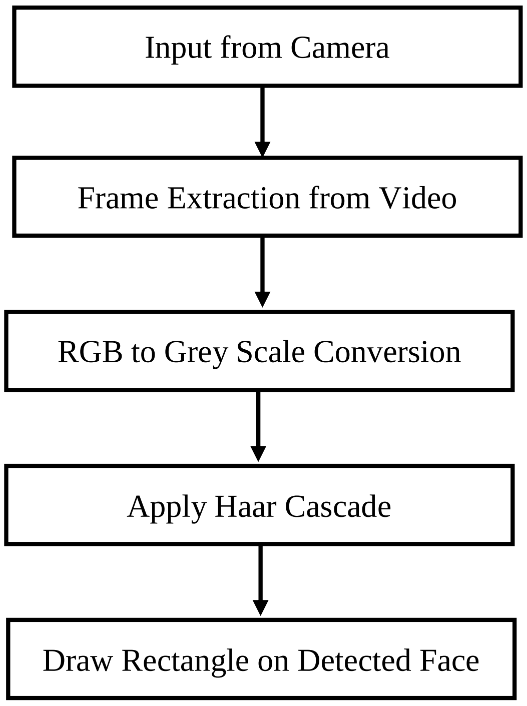

Input from Camera
Frame Extraction from Video
RGB to Grey Scale Conversion
Apply Haar Cascade
Draw Rectangle on Detected Face

Figure 4.2: Block Diagram of Face Detection

The *detectmultiscale()* function of the cascade classifier returns the bounding boxes for all the faces present in the current frame. RGB frame is converted to grey scale. General step is to scan the whole image in given frame for the faces. This process often returns un-necessary bounding rectangles on non-face images. Therefore, we implemented face detection using OpenCV pretrained XML detector in our final application. OpenCV comes with some pretrained Haar-cascade detectors   to use for face detection. The cascade classifier XML filename used was *haarcascade_frontalface_default.xml*. The region of interest i.e. detected faces in frame is passed to preprocessing stage and then to prediction model to predict decision on the detected faces.

Different parameters passed on *detectmultiscale*() are:

- MinFeatureSize: This parameter determines the minimum face size that we care about, typically 20 x 20 or 30 x 30 pixels but this depends on our use case and image size.
- SearchScaleFactor: The parameter determines how many different sizes of faces to look for; typically it would be 1.1 for good detection, or 1.2 for faster detection.
- MinNeighbors: This parameter determines how sure the detector should be that it has detected a face, typically a value of 3 but we can set it higher if we want more reliable faces, even if many faces are not detected.

4.3.4. Preprocessing

Face images are pre-processed and enhanced to improve the recognition performance of the system. Based on the requirement, different preprocessing techniques related to face recognition process are described with the help of block diagram as follow:

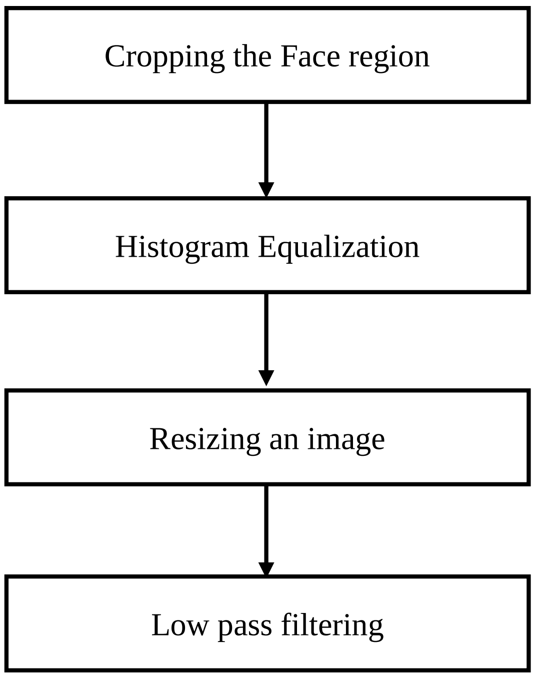
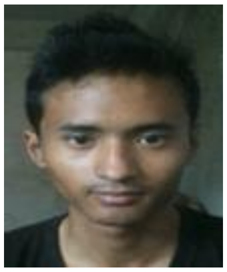


Figure 4.3: Preprocessing Block Diagram

4.3.4.1. Cropping Face Region

The region of the image where the face is located is cut out from the image and only this area is used in the process of face recognition. By using cropping technique only main face can be extracted and the redundant data around the face, which deteriorates the performance of recognition can be removed.


(i)

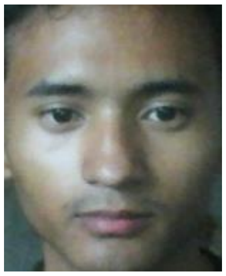

(ii)

Figure 4.4: (i) Original image and (ii) Cropped image

4.3.4.2. Histogram Equalization

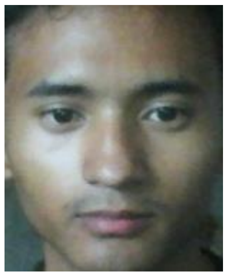
Histogram equalization was done in order to enhance image quality and to improve face recognition performance. It changes the dynamic range (contrast range) of the image and as a result, some important facial features become more visible. Mathematically histogram equalization can be expressed as:

$$S_k = T(r_k) = \sum_{i=0}^{k} \frac{n_i}{n} \qquad (4.1)$$

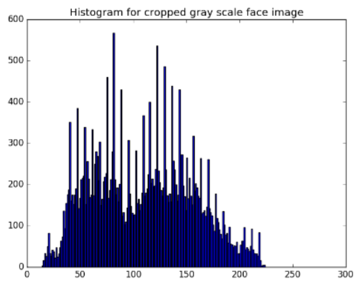
Where k= 0, 1, 2, …. ,L-1

Here in equation (3.1), ‘n’ is the total number of pixels in an image, $'n_j'$ is the number of pixels with gray level ‘r_k’, and ‘L’ is the total number of grey levels exist in the face image.

The above cropped image and its histogram is shown side by side as:



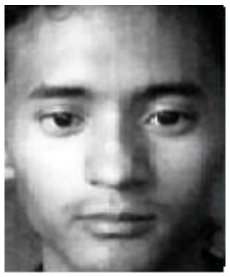
(i)



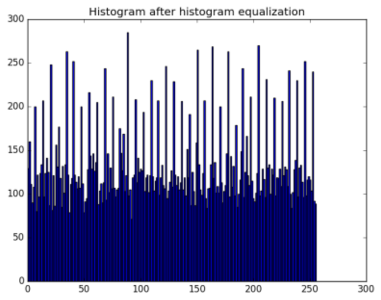
(ii)

Figure 4.5: (i) Cropped Image and (ii) histogram for cropped image

The above histogram hides many discriminative features so to enhance the features the histogram equalization must be done. The cropped image after histogram equalization is shown below:



(i)



(ii)

Figure 4.6: (i) Histogram equalized image and (ii) Histogram after equalization

4.3.4.3. Image Resizing

The process of image resizing changes the size of the image. The size of the image is scaled down to reduce the resolution of face image. This reduction in size is done to reduce mathematical complexity in LDA and KNN process .For resizing face image, different interpolation techniques exist among them Bi-cubic interpolation method was used in the proposed work, the advantage of resizing through Bi-cubic interpolation was that, it produces more smoother surfaces than any other interpolation technique. In Bi-cubic Interpolation technique, it considers 16 neighboring pixels in the rectangular grid and calculates weighted average of these pixels to replace them with a single pixel, it is that pixel, which has got the flavor of all the 16 replaced pixels.

4.3.4.4. Low Pass Filtering

The Gaussian filter was applied in order to produce the blurry effect, because in the later stages face recognition algorithm include step of face image re-sizing while maintaining the quality of face image. The 5 X 5 filter is used for this process.

$$R = \frac{1}{25}\sum_{i=1}^{25} Z_i \tag{4.2}$$

Equation (4.2) calculates the average value of the pixels, whereas 'z' is the mask, 'i' are mask elements. The mask is then convolved with image to produce filtering effect, for a 5 X 5 mask used in the implementation, it calculate the average of 25 pixels in that filter mask.

4.3.5. Machine Learning

The preprocessed training data sets of face images were represented into a list of numpy arrays with corresponding labels i.e. the unique number of the person in the list. Basically all face recognition algorithms are the combination of a feature extraction and a classifier. Subspace-LDA is used as feature extraction method and K-NN classifier with Euclidean-Distance is used as classifier .The feature and classifier form a model which we called PredictableModel does feature extraction and learns the classifier. Once we have created our model we learn it on our given test data and their labels. This completes training of prediction model on the set of images we have prepared. The classifier always outputs a list with corresponding predicted label and confidence based on value of K used. The learnt model was saved so that we can easily reuse the model and don't need to learn it from the dataset all over again.

4.3.6. Face Recognition

Testing the performance of our recognition system is done on preprocessed test datasets by loading model generated using training dataset. Features are extracted form input test image and compared with saved model to select a class with maximum likelihood to which input image belongs and provides predicted label from which system can easily predict to which image/person in training set it actually belongs to. Testing was initially done on ORL face database i.e. still images .After finding desirable performance of model on this database, we used this model with our IOE dataset for final implementation of our desired system.

When new image not in dataset come for recognition it is recognized with training image with lowest score. To solve this problem we need a threshold value. For setting the threshold value we calculated distances for features vector of people in dataset and for random set and set the threshold .We pass this decision threshold to predict method, which a prediction is threshold against.

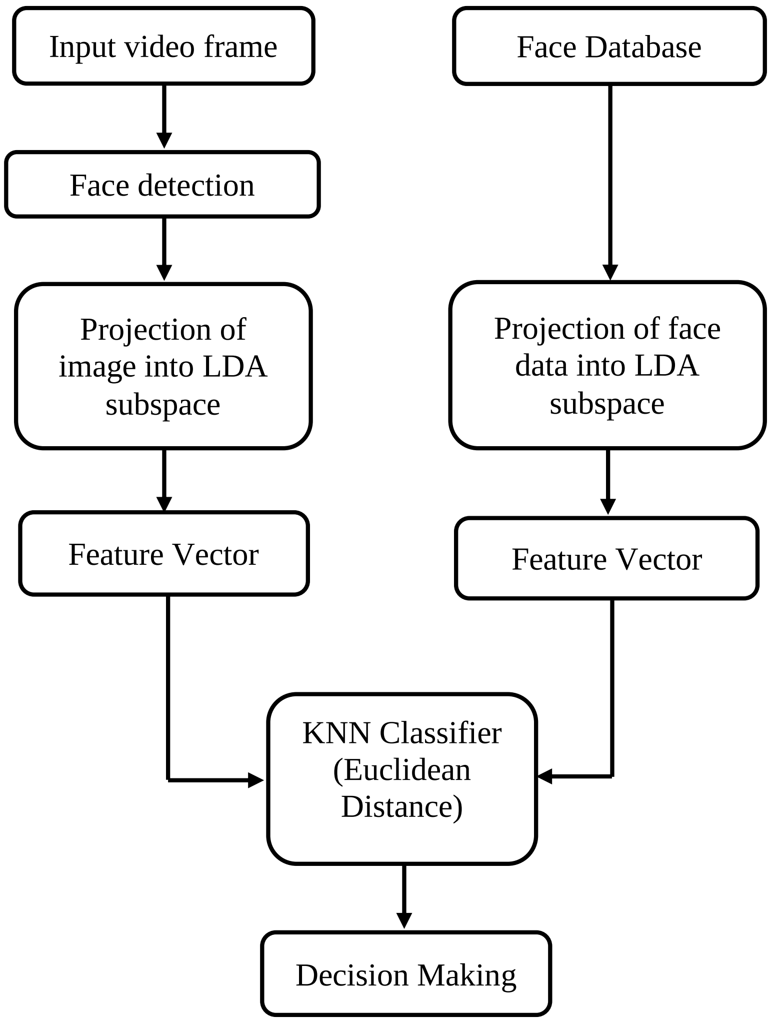

Input video frame
Face Database
Face detection
Projection of image into LDA subspace
Projection of face data into LDA subspace
Feature Vector
Feature Vector
KNN Classifier (Euclidean Distance)
Decision Making

Figure 4.7: Face Recognition Flowchart

4.3.7. User Interface

Web application developed using .net framework has been used as user interface. The web application has been used to provide accessibility to the users. It has been developed to authenticate the user via register and login webpages. It provides result of the python application in a generated result webpage in response to user's query taken with a search webpage.

5. Performance and Analysis

The face recognition module was tested on some standard face databases and the performance of the face extraction module was tested on a video stream by webcam at 640 × 480 resolution. Testing was performed on a Dell laptop with the following specifications:

- Processor: Intel(R) Core(TM) i5-3337U CPU @ 1.8 GHz
- Memory: 4 GB RAM
- Operating System: Windows 8.1

All the performance related data presented here relate to natural lighting condition, almost no facial occlusion, large amount of facial expression variation and very small out of plane rotation (10^0) face.

5.1. Result of Face Detection

The detection on the webcam images are better than the images selected at random because the non-face source images used for the training was of the environment on which the detector would be run. The detections of other images are good for the near images, but for other images there are many false detections. This can be improved by training the detector on a larger training set and training the detector for larger stage. The face detection using AdaBoost is shown below:

Table 1: Face Detection Count

Input video stream	Actual Face Count	Detected Face Count	True Detect Count
1.	2	2	2
2.	4	6	3
3.	6	6	4
4.	8	5	4

The overall accuracy of face detection was found to be 72.75%.

The detection rate was higher for the video frames in which the face of the subject was directly facing the camera and when the video frame contained only one subject. There are higher false detections in many video frames because the cascade was trained only up to 10 stages (due to long time required for the training). So, for reducing the number of false positives, the detector has to be trained to higher stages.

5.2. Result of Face Recognition

5.2.1. Performance of Subspace LDA on ORL Database

The recognition rate on ORL Database for different number of training set (Out of ten face images from each subject, seven and five were taken as a training set) with different values of K for KNN classifier. With increased number of training set, the recognition rate also increases. The result for different values of K is shown below:

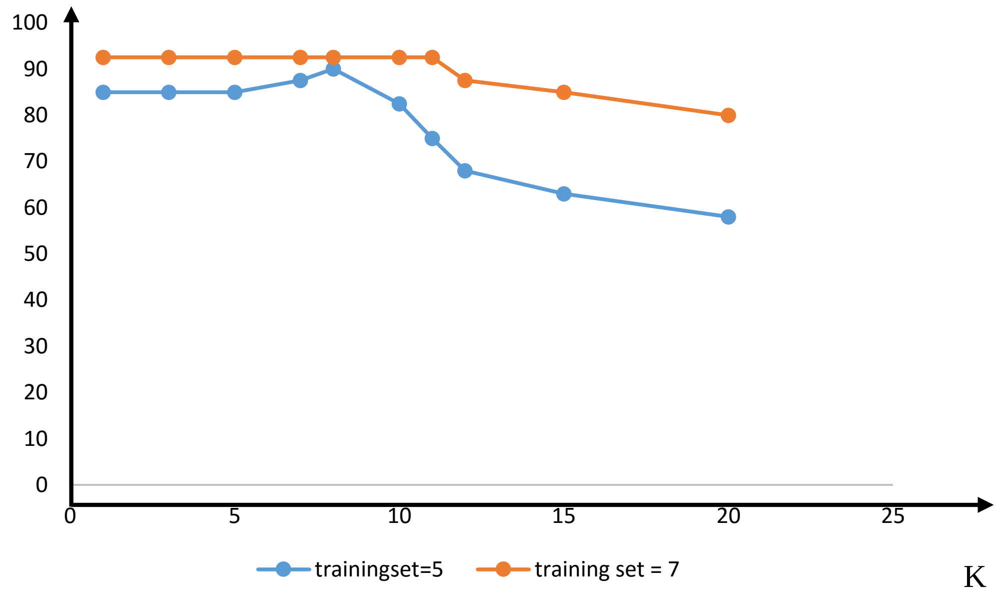
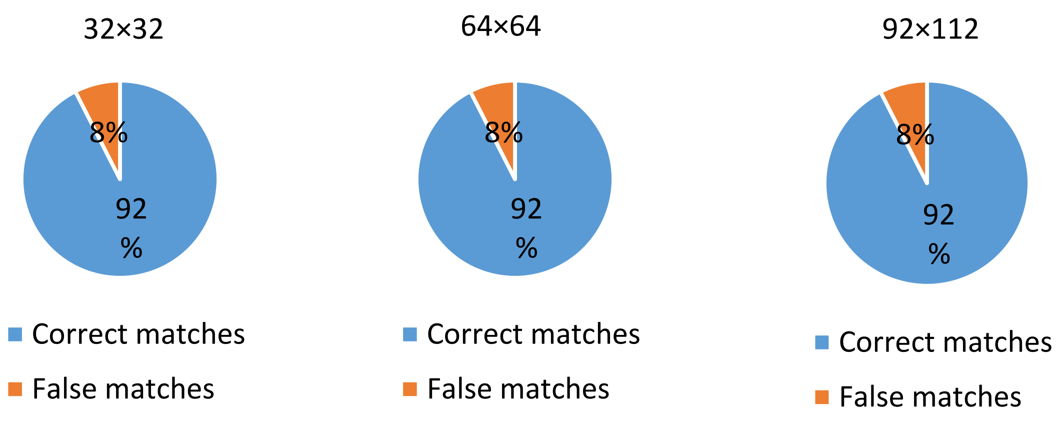


Figure 5.1: Recognition rate on ORL Database

The best result obtained with all image sizes i.e. 32 ×32, 64×64, 92×112 was correct match of 92% and 8% of false match. The result below shows the performance of Subspace LDA on ORL Database:



(i) (ii) (iii)

Figure 5.2: Recognition rate for subspace LDA on ORL Database for train/probe images of three different sizes (i) for 32x32, (ii) for 64x64, (iii) for 92x112

5.2.2. Performance of Subspace LDA on IOE Thapathali database

The best result obtained with all image sizes i.e. 32 ×32, 64×64, 92×112 was correct match of 90% and 10% of false match. The result below shows the performance of Subspace LDA on IOE Thapathali Database was:

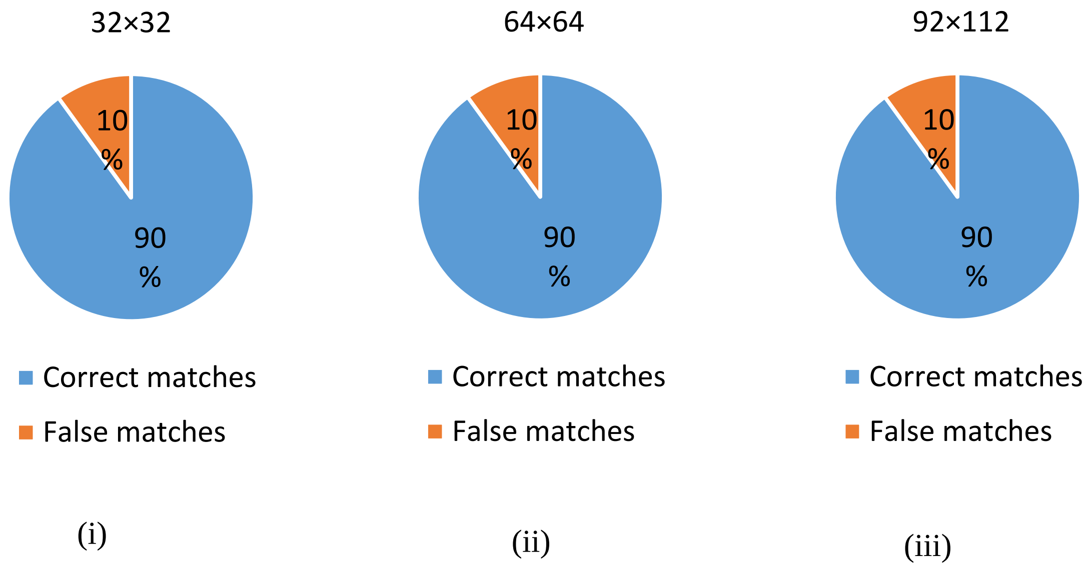


Figure 5.3: Recognition rate for subspace LDA on IOE Thapathali database for train/probe images of three different sizes (i) for 32x32, (ii) for 64x64, (iii) for 92x112

Subspace LDA, being a holistic approach, is not affected by the resolution of test images means it can recognize faces as long as the overall facial structure is preserved. This is a remarkable property of Subspace LDA which can be used to recognize faces in a video stream. However, the performance of Subspace LDA algorithm degrades when there is considerable illumination variation between the training and test face images or when profile faces (side looking faces) are present in the test set.

6. Limitations and Further Enhancement

6.1. Limitations

The limitations of system are as follows:

- Same resolution camera has to be used for collection of training datasets and that for the final application.
- Under different lighting illumination an unknown person i.e. person not in datasets is wrongly recognized to be one in the datasets.
- System is limited to webcam and a USB camera.
- Difficulty in placement and orientation of camera for wide range
- The designed system has been developed only as a web application for desktop and laptop (no access for mobile user).
- System is not rotation invariant.
- Single user system where at any time only a single user can interact with system and only single person can be searched at that time.
- Algorithms are best chosen after effective comparison among acquired ones. Nevertheless, PCA, LDA and KNN lacks cent percent accuracy.

6.2. Further Enhancement

There is a great possibility to enhance this project in upcoming future.

- The performance of face detection using haar-like classifier created through haar training process can be improved by increasing number of positive and negative images for training.
- The feature extraction and classification algorithm has the greatest possibility of being enhanced so that system becomes rotation invariant and can perform well under different lighting illumination thus increasing the accuracy.
- The current system integrates webcam and a USB camera only. The addition of number of cameras for extension of coverage range makes system more effective and real time. Similarly integration of number of wireless or IP camera makes system

more efficient and user interactive in which user can get access the system from distant places.

- People now are more attracted towards Android usage. The extension of this system towards Android platform deliberately improves users' capability to access the system on their palm.

7. Conclusion

As the system aims for many objectives during its development purpose, the actual instances for the design and analysis of various system components and model really worked and deliberately fulfills most of its objectives from face detection to face recognition. The development of the proposed system started with the research and collection of data for detection and recognition. Human face detection is often the first-step in the recognition process as detecting the location of a face in an image, prior to attempting recognition can focus computational resources on the face area of the image. For face detection viola johns object detection framework has been used .For recognition task, PCA is used for feature/dimension reduction, LDA for feature extraction and KNN for classification. The system has been tested on a wide variety of face images from ORL and IOE datasets which is giving desirable accuracy.

A system that has been developed finds location of concerned persons within the datasets based on face recognition. The system is capable of using webcam and a camera as the capturing device simultaneously and providing the detail location information of person whether he/she is found or not. If person is recognized as one in the datasets then the output which is the location of camera from which that person has been recognized is provided to the user as final result. User interaction with system is done through web application developed using ASP.NET MVC framework. Thus, the system is good and efficient for location detection of person in any educational institutes, offices or any industries etc. We summarize the progress with respect to the main objectives of the project, namely, accuracy and consistency.

- Accuracy: Accuracy is the main gist of the project. The accuracy of face detection is around 72.75% and face recognition is near about 90%, therefore we are able to obtain satisfactory result in face recognition.

- Consistency: Consistency is also a tedious task for this project. The requirement for decrease in the inconsistent result has made it difficult to balance between accuracy and consistency. However, by the use of the image processing techniques, we have been able to improve the consistency resulting in the consistent output of the system.

8. REFERENCES

[1] A. Jain, A. Ross, S. Prabhakar, "An Introduction to Biometric Recognition", 2004.

[2]A. Kalkanova, "Comparison of Various Biometric Method", University of Southampton.

[3] A.S. Tolba, A. El-Baz, and A.A. El-Harvy, "Face Recognition: Literature Review", International Journal of Signal Processing, 2006.

[4] S. Bajpai, A.Singh, and K.V. Karthik, "An Experimental Comparison of Face Detection Algorithm", International Journal of Computer Theory and Engineering, 2013.

[5] P. Voila and M.J. Jones, "Robust Real Time Face Detection", International Journal of Computer Vision, 2004.

[6] P. Voila, M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", 2001.

[7] Constantine P. Papageorgiou, Michael Oren and Tomaso Poggio, "A General Framework for Object Detection", Proceedings of the 6th IEEE International Conference on Computer Vision, 1998.

[8] Yoav Freund and Robert E. Schapire, "Experiments with a New Boosting Algorithm", Proceedings of the 13th International Conference on Machine Learning, 1996.

[9] Yi-Shin Liu Wai-Seng Ng Chun-Wei Liu, "A Comparison of Different Face Recognition Algorithms", National Taiwan University.

[10] Juwei Lu, K.N. Plataniotis, and A.N. Venetsanopoulos, "Face Recognition Using LDA Based Algorithms".

[11] K.Etemad, R.Chellappa,"Discriminant analysis for recognition of human face images", 1997.

[12] Li-Fen Chen, Hong-Yuan Mark Liao, Ming-Tat Ko, Ja-Chen Lin, Gwo-Jong Yu : "A new LDA-based face recognition system which can solve the small sample size problem", Pattern Recognition 33 (2000) 1713-1726.

[13] K. Fukunaga, "Introduction to Statistical Pattern Recognition", Academic Press, New York, 1990.

[14] Kilian Q. Weinberger, John Blitzer and Lawrence K.Saul, " Distance Metric Learning for Large Margin Nearest Neighbor Classification", 2005.

[15] R.C. Gonzalez and R.E. Woods, “Digital Image Processing”, Second Edition, Prentice Hall.

[16] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, December 2001.

[17] Bo Wu, Haizhou Ai, Chang Huang, “Fast Rotation Invariant Multi-View Face Detection based on Real Adaboost”, International Conference on Automatic Face and Gesture Recognition, 2004.

[18] Peter Belhumeur N., Joao Hespanha P., and Kriegman J. David, “Eigenfaces vs. fisherfaces: Recognition using class specific linear projection”, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 19(7), 1997.

[19] Rainer Leinhart and Jochen Maydt, “An Extended Set of Haar-like Features for Rapid Object Detection,” Intel Labs, Intel Corporation, Santa Clara, CA 95052, USA, Sep. 2002.

[20] W. Zhao, R. Chellappa, and P. J. Phillips, “Subspace linear discriminant analysis for face recognition”, Technical Report CAR-TR-914, Center for Automation Research, University of Maryland, College Park, MD., 1999.

[21] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, “Face recognition: A literature survey”, In ACM Computing Survey, volume 35, pages 399–458, December 2003.

[22] A. Mohan, C. Papageorgiou, and T. Poggio, “Example-based object detection in images by components”, IEEE Trans. Pattern Anal. Mach. Intell., 23(4):349–361, 2001.

Appendix A: Project Output

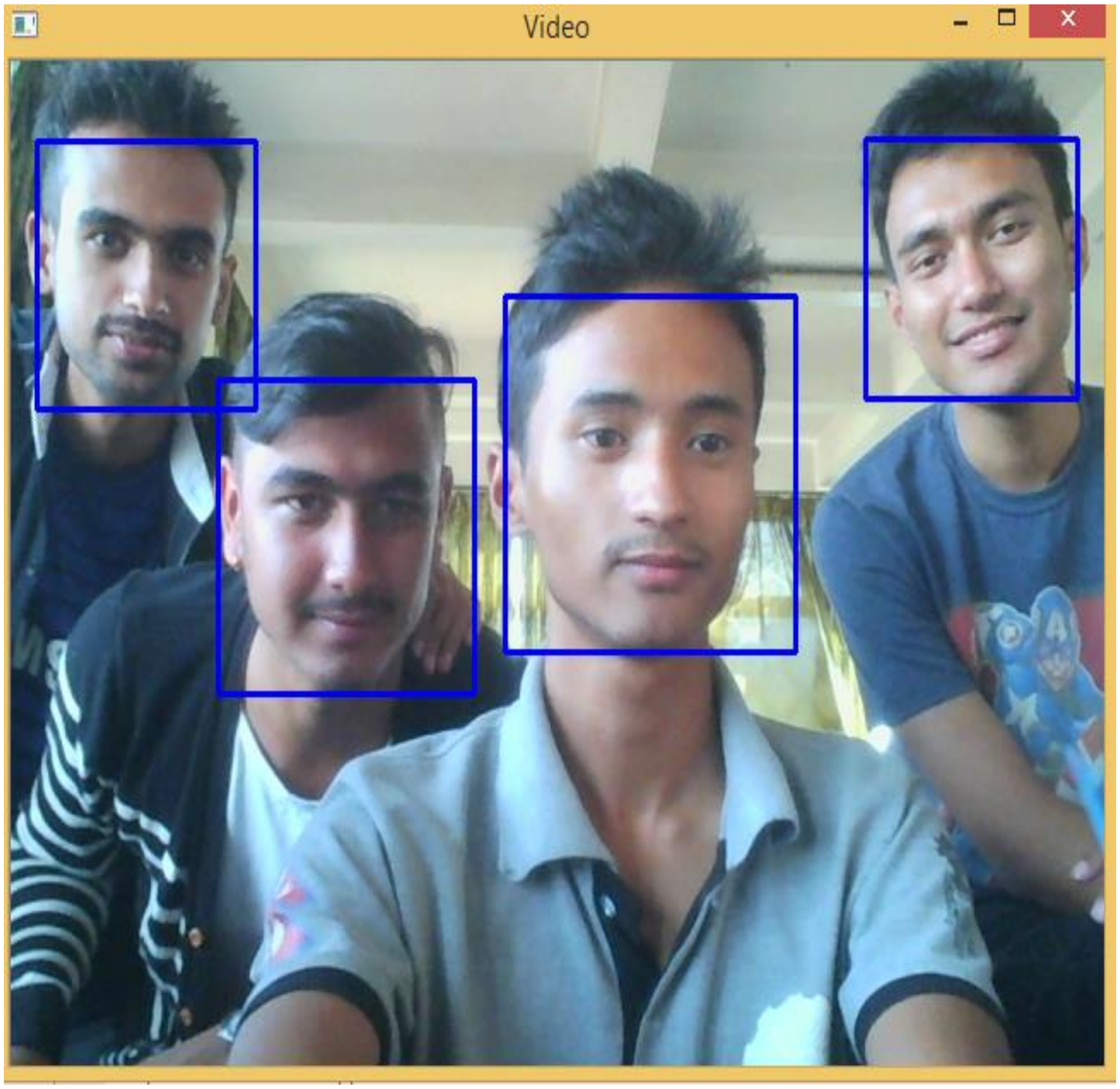


Figure 1: Window showing face detection of persons

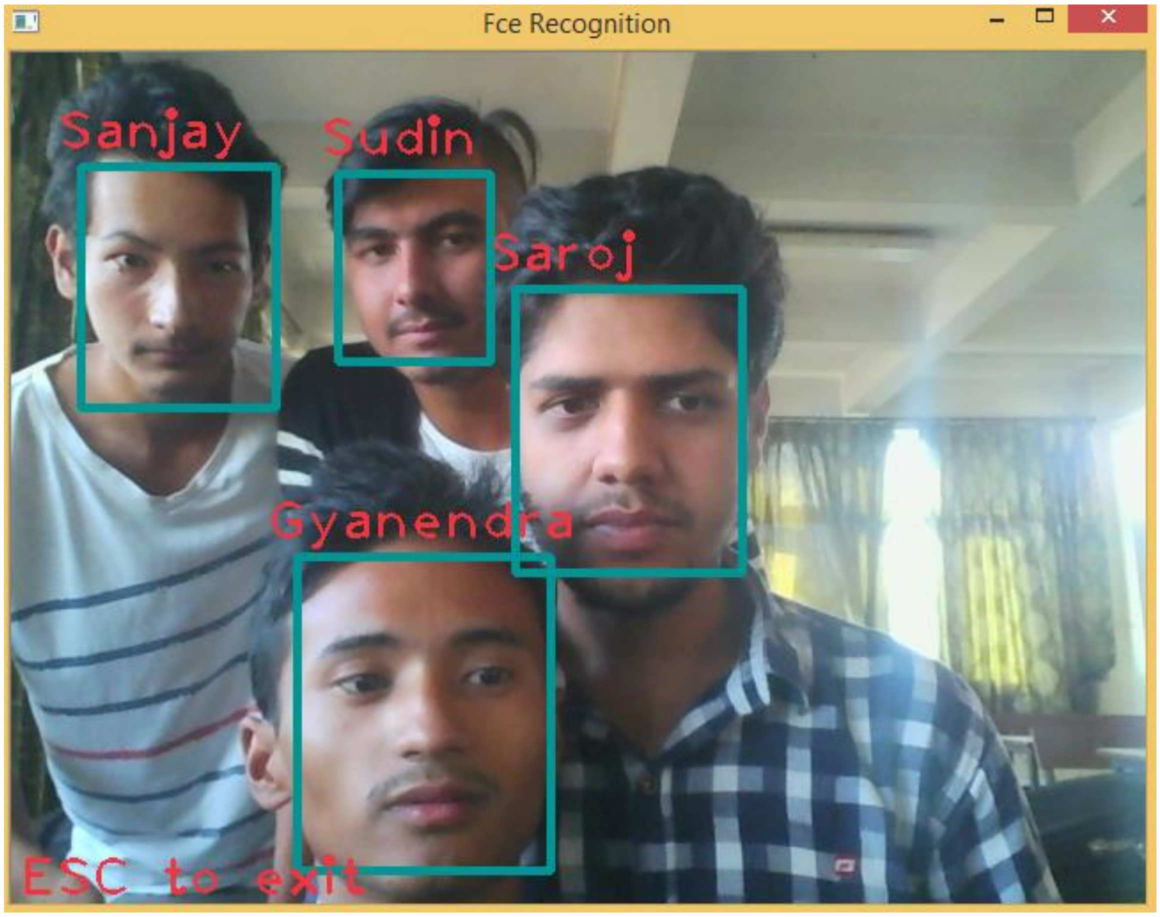


Figure 2: Window showing face recognition of persons

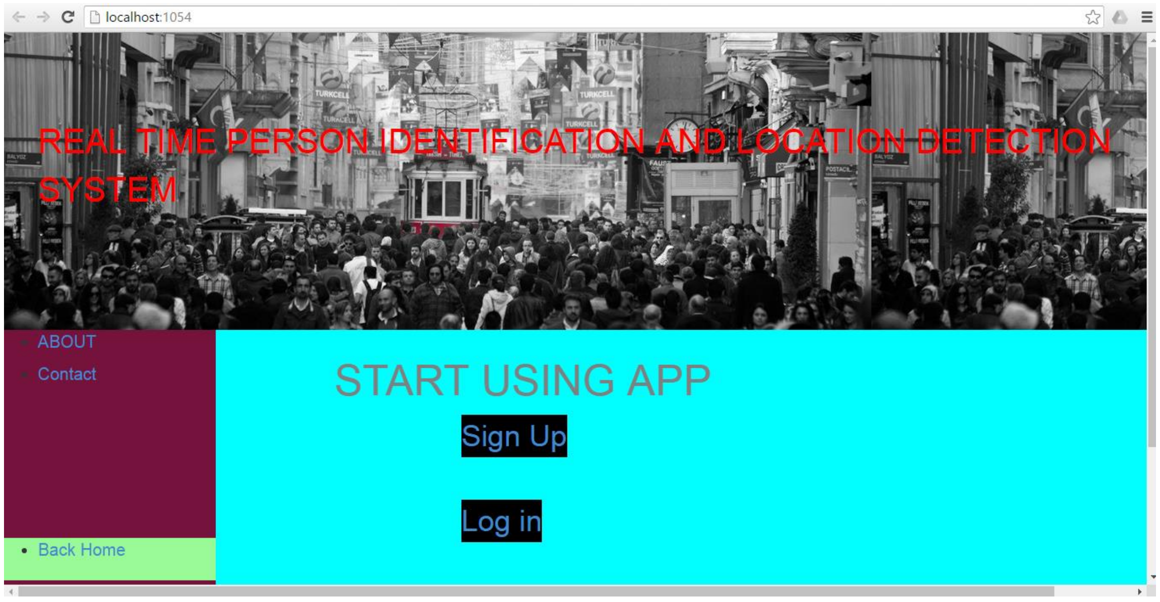


Figure 3: Start up screen of web application

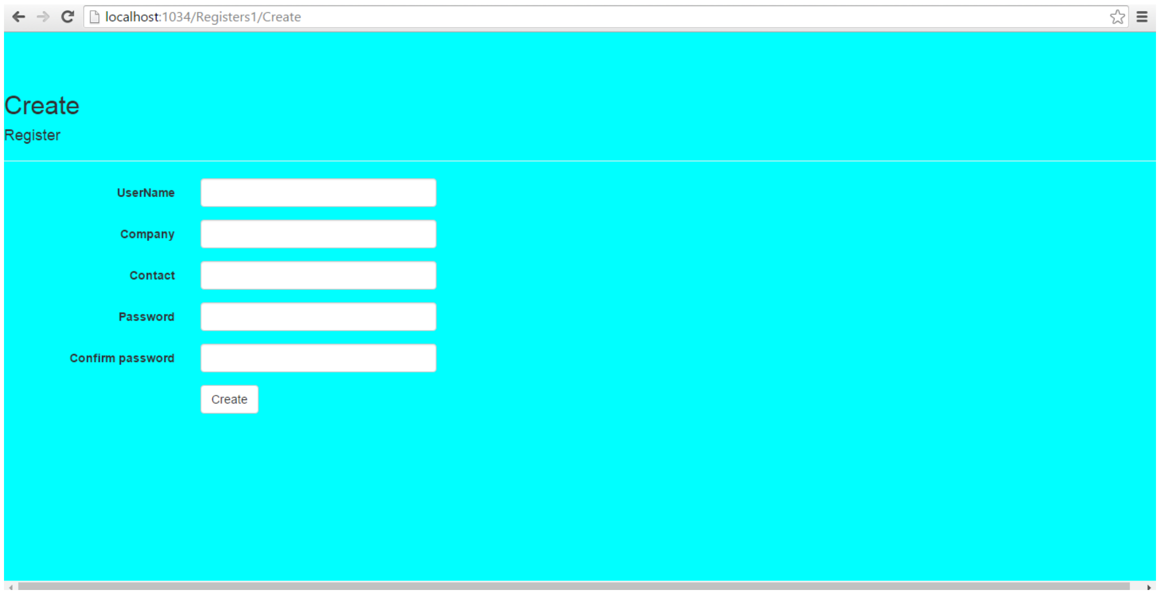
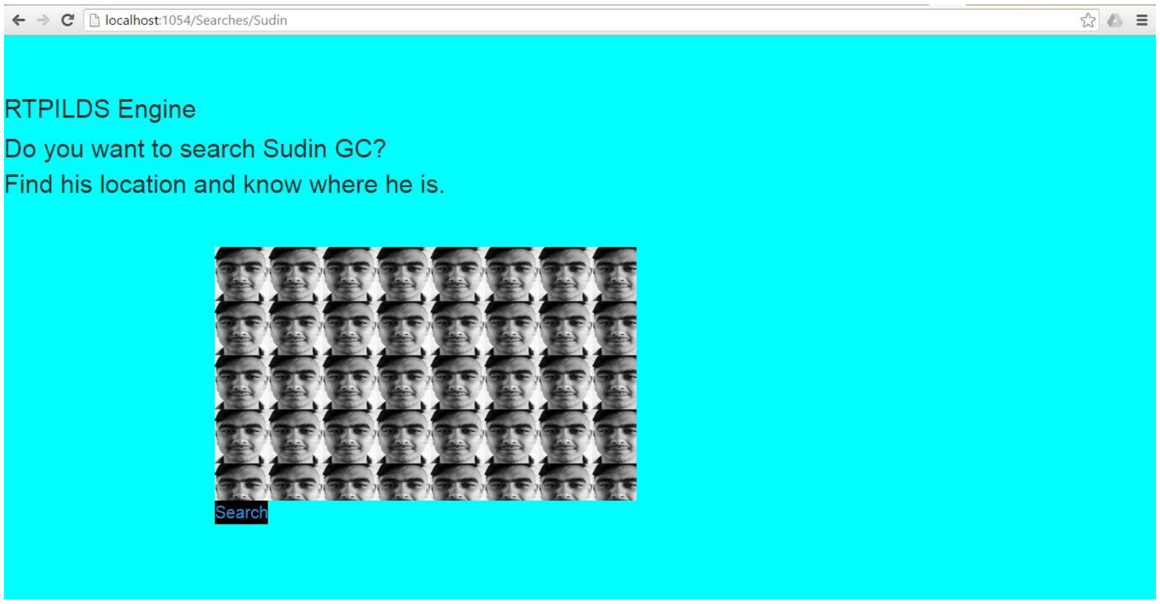


Figure 4: Window showing the register page



RTPILDS Engine

Do you want to search Sudin GC?

Find his location and know where he is.

Figure 5: Window showing the query for searching

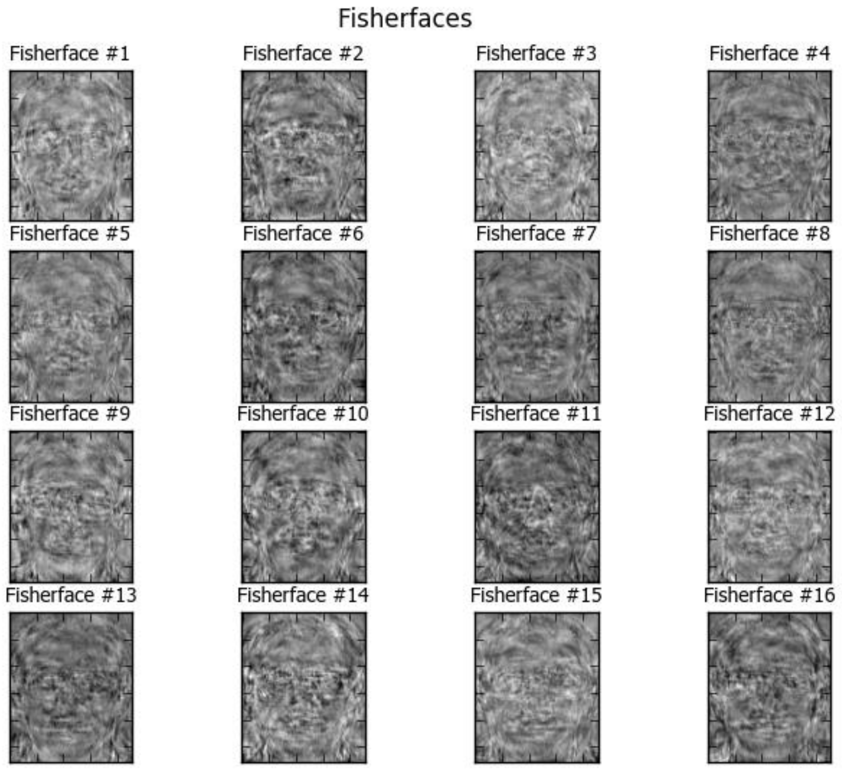


Figure 6: Fisherfaces for ORL Database

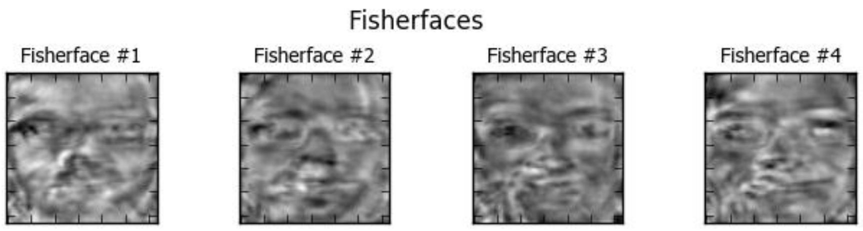


Figure 7: Fisherfaces for IOE Database

Appendix B: Setup Manual

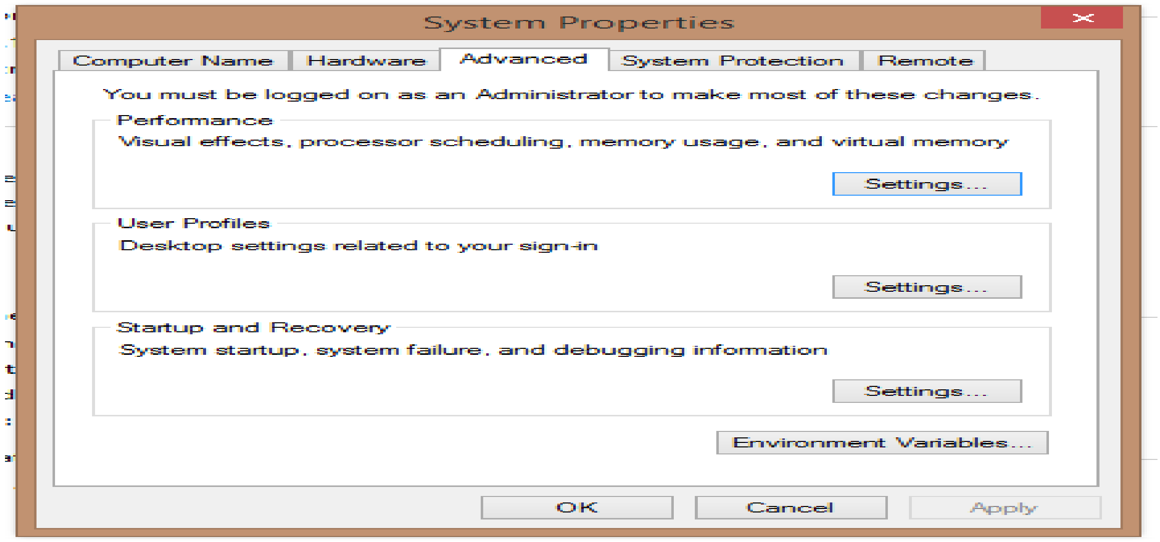
System Requirements:

1. Python 2.7
2. SQL
3. OS (Windows 32/64 bit)
4. Visual Studio
5. Web browser (Google Chrome, Mozilla Firefox….)

Python 2.7 setup:

- Download Python 2.7 from, https://www.python.org/download/releases/2.7/
- Install python 2.7 IDE PyCharm IDE from, https://www.jetbrains.com/pycharm/download/

  Install python and give it administrative right as follows:



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

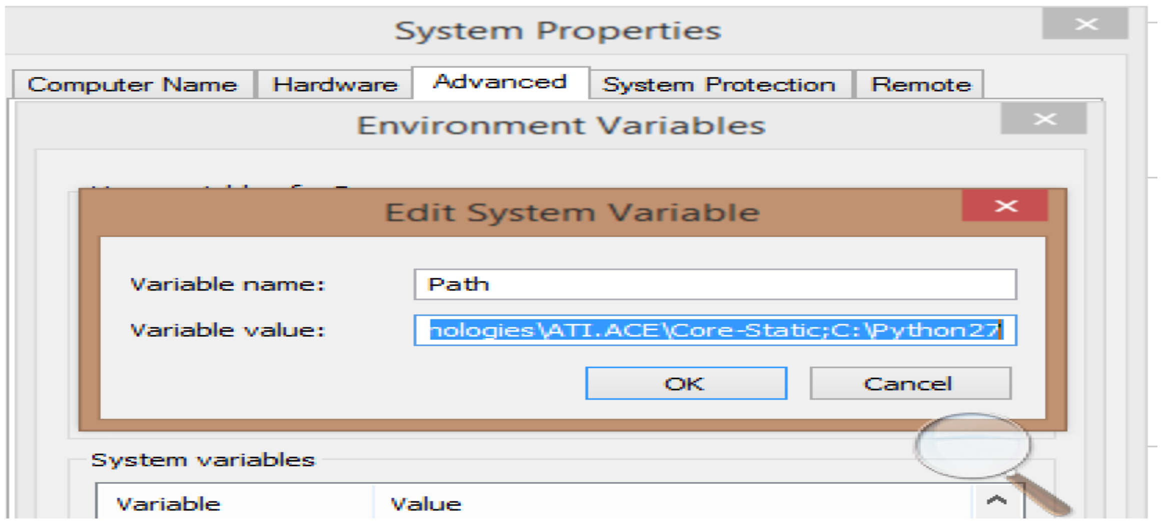
User Profiles
Desktop settings related to your sign-in
Settings...

Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK | Cancel | Apply

- Go to System Properties of your PC through the advanced setting,
- Click on Advanced tab. Then click on Environment Variables.



Variable name: Path

Variable value: nologies\ATI.ACE\Core-Static;C:\Python27

- Above figure is displayed, then click on path and edit. Then add the location of your python installation to the "Variable value". Hit Save.
- Now you are able to access python through command prompt having administrative privileges. This is necessary because to install modules or packages of python through command prompt, administrative privileges for python is required.

Essential python libraries

- Numpy
- OpenCV
- Matplotlib

Web application startup

- Start Browser
- Browse http://localhost:<port_number_used_by_IIS Server>

Appendix C: Face Database

Folder structure of ORL face database

```
|-- README
|-- s1
|   |-- 1.pgm
|   |-- 2.pgm
[...]
|   `-- 10.pgm
|-- s2
|   |-- 1.pgm
|   |-- 2.pgm
[...]
|   `-- 10.pgm
|-- s3
|   |-- 1.pgm
|   |-- 2.pgm
[...]
|   `-- 10.pgm

..40 directories, 401 files
```

Folder Structure of IOE face database

```
|-- Sudin

|    |-- 1.jpg

|    |-- 2.jpg

|    |-- 3.jpg

|    |-- 4.jpg

|-- Sanjaya
```

```
|   |-- 1.jpg

|   |-- 2.jpg

|   |-- 3.jpg

|   |-- 4.jpg



[...]
```

Log file for haar training

```
/usr/bin/python2.7 /home/sudin/PycharmProjects/untitled/create_sample.py
PARAMETERS:
cascadeDirName: data
vecFileName: positives.vec
bgFileName: bg.txt
numPos: 20000
numNeg: 8000
numStages: 10
precalcValBufSize[Mb] : 256
precalcIdxBufSize[Mb] : 256
stageType: BOOST
featureType: HAAR
sampleWidth: 20
sampleHeight: 20
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
```

Log file for end stage

```
Home/sudin/PycharmProjects/haartraining/create sample.py"
PARAMETERS:
cascadeDirName: data
vecFileName: positives.vec
bgFileName: bg.txt
numPos: 20000
numNeg: 8000
numStages: 10
precalcValBufSize[Mb] : 256
precalcIdxBufSize[Mb] : 256
stageType: BOOST
featureType: HAAR
sampleWidth: 20
sampleHeight: 20
boostType: GAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: BASIC

Stages 0-7 are loaded
===== TRAINING 8-stage =====
<BEGIN
POS count : consumed   20000 : 20400
NEG count : acceptanceRatio    8000 : 0.0599215
Precalculation time: 9
END>
Training until now has taken 2 days 4 hours 1 minute 27 seconds.
```