GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

AUTOMATED ATTENDANCE SYSTEM USING FACE RECOGNITION

A Project Report of Capstone Project - 2

Submitted by

SALMAN HUSAIN

(15SCSE111002)

VISHWA MOHAN

(18SCSE2030049)

in partial fulfilment for the award of the degree

of

MASTER OF COMPUTER APPLICATION

IN

COMPUTER AND INFORMATION SCIENCES

SCHOOL OF COMPUTER AND INFORMATION SCIENCES

Under the Supervision of

Mr. Ajay Kumar (Assistant Professor)

MAY - 2020

GALGOTIAS UNIVERSITY

SCHOOL OF COMPUTER AND INFORMATION SCIENCES

BONAFIDE CERTIFICATE

Certified that this project report "Automated Attendance System Using Face Recognition" is the bonafide work of "SALMAN HUSAIN AND VISHWA MOHAN" who carried out the project work under my supervision.

SIGNATURE OF HEAD	**SIGNATURE OF SUPERVISOR**
Dr. MUNISH SHABARWAL, **PhD (Management), PhD (CS)** **Professor & Dean,** **School of Computer and informat-** **-ion Sciences**	**Mr. Ajay Kumar Assistant** **Professor** **School of Computer and information** **Sciences.**

ABSTRACT

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity. Human face recognition procedure basically consists of two phases, namely face detection, where this process takes place very rapidly in humans, except under conditions where the object is located at a short distance away, the next is the introduction, which recognize a face as individuals. Stage is then replicated and developed as a model for facial image recognition (face recognition) is one of the much-studied biometrics technology and developed by experts. There are two kinds of methods that are currently popular in developed face recognition pattern namely, Eigenface method and Fisher face method. Facial image recognition Eigenface method is based on the reduction of face-dimensional space using Principal Component Analysis (PCA) for facial features. In this project face detection and face recognition is used. Face detection is used to locate the position of face region and face recognition is used for marking the understudy’s attendance. The database of all the students in the class is stored and when the face of the individual student matches with one of the faces stored in the database then the attendance is recorded.

TABLE OF CONTENTS

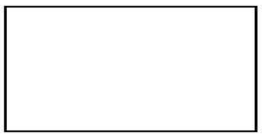
List of Figures

LIST OF ABBREVIATIONS

Abbreviations	Description
CSV	Comma Separated Values
DFD	Data Flow Diagram
GTK	Graphical User Interface Toolkit
GUI	Graphical User Interface
OpenCV	Open Source Computer Vision
I/O	Input/ Output
SMS	Short Message Service
ID	Identification
DD	Data Dictionary
RGB	Red Green Blue
LBPH	Local Binary Patterns Histograms
PDM	Point Distribution Model
FRC	Fingerprint Based Recognition System
PCA	Principle Component Analysis
RBF	Radial Basis Function
RFID	Radio Frequency Identification
MATLAB	Mat rix Lab oratory
LDA	Linear Discriminate Analysis
IRS	Iris Based Recognition System
SVM	Support Vector Machine
YML	YAML Ain't Markup Language
CCTV	Closed Circuit Television
PDBNN	Probabilistic Decision Based Neural Network
EBGM	Elastic Bunch Graph Map
HOG	Histogram of oriented gradient
FRS	Face Based Recognition System
ER Diagram	Entity Relationship Diagram

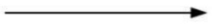
Notation

External Entity



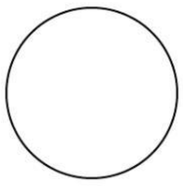
External entities are objects outside the system, with which the system communicates. External entities are sources and destinations of the system's inputs and outputs

Data Flow



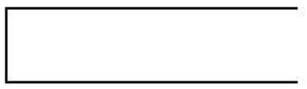
Dataflow are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.

Process



Transform of incoming data flow(s) to outgoing flow(s).

Data Store



Data stores are repositories of data in the system. They are sometimes also referred to as files, queue or as sophisticated as a relational database

Naming and Convention

Principal Component Analysis (PCA)

In simple words, principal component analysis is a method of extracting important variables (in form of components) from a large set of variables available in a data set.

Local binary patterns (LBP)

It is a type of visual descriptor used for classification in computer vision which is found to be a powerful feature for texture classification; it has further been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Eigen Face

Eigenfaces is the name given to a set of eigenvectors when they are used in the computer vision problem of human face recognition.

Viola–Jones

The Viola–Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time. Which can be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection.

Haar Features

Haar-like features are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector.

Ad boost

Ad boost, short for Adaptive Boosting, is a machine learning meta-algorithm which can be used in conjunction with many other types of learning algorithms to improve performance by 50%.

Classifier

An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category.

Cascading

Cascading is a case of ensemble learning based on the concatenation of several Classifiers, using all information collected from the output from a given classifier as additional information for the next classifier in the cascade.

Linear Discriminant Analysis (LDA)

Linear discriminant analysis (LDA) is a generalization of Fisher's linear discriminant, a method used in statistics, pattern recognition and machine learning to find a linear combination of features that characterizes or separates two or more classes of objects or events.

Cascade Object Detector

The Computer Vision System Toolbox cascade object detector can detect object categories whose aspect ratio does not vary significantly which comes with several pretrained classifiers for detecting frontal faces, profile faces, noses, eyes, and the upper body.

InsertObjectAnnotation

RGB = insertObjectAnnotation (I, shape, position, label, Name, Value) uses additional options specified by one or more name-value pair arguments. Example: - insertObjectAnnotation (I, ‘rectangle', position, label) inserts rectangles and labels at the location indicated by the position matrix.

Computer Vision Toolbox

Design and simulate computer vision and video processing systems which provides algorithms, functions, and apps for designing and simulating computer vision and video processing systems.

Image Acquisition Toolbox

Image Acquisition Toolbox enables you to acquire images and videos from cameras and frame grabbers directly into MATLAB and Simulink.

Image Processing Toolbox

Image Processing Toolbox provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development.

CHAPTER 1

Introduction

1.1 Background Introduction

The current method that colleges use is that the faculty passes a sheet or make roll calls and mark the attendance of the students and this sheet further goes to the admin department with updates in the final excel sheet. This process is quite hectic and time consuming. Also, for professors or employees at institutes or organizations the biometric system serves one at a time. So, why not shift to an automated attendance system which works on face recognition technique? Be it a classroom or entry gates it will mark the attendance of the students, professors, employees, etc.

1.2 Using Biometrics

Biometric Identification Systems are widely used for unique identification of humans mainly for verification and identification. Biometrics is used as a form of identity access management and access control. So, use of biometrics in student attendance management system is a secure approach. There are many types of biometric systems like fingerprint recognition, face recognition, voice recognition, iris recognition, palm recognition etc. In this project, we used face recognition system.

1.3 Motivation

The main motivation for us to go for this project was the slow and inefficient traditional manual attendance system. These made us to think why not make it automated fast and much efficient. Also, such face detection techniques are in use by department like crime investigation where they use CCTV footages and detect the faces from the crime scene and compare those with criminal database to recognize them.

1.4 Current Systems

At present attendance marking involves manual attendance on paper sheet by professors and teachers but it is very time-consuming process and chances of proxy is also one problem that arises in such type of attendance marking. Also, there are attendance marking system such as RFID (Radio Frequency Identification), Biometrics etc. but these systems are currently not so much popular in schools and classrooms for students.

1.5 The Problems with Current System

The problem with this approach in which manually taking and maintains the attendance record is that it is very inconvenient task. Traditionally, student attendances are taken manually by using attendance sheet given by the faculty members in class, which is a time-consuming event. Moreover, it is very difficult to verify one by one student in a large classroom environment with distributed branches whether the authenticated students are responding or not. The ability to compute the attendance percentage becomes a major task as manual computation produces errors and wastes a lot of time. This method could easily allow for impersonation and the attendance sheet could be stolen or lost.

1.6 Existing System

1. Fingerprint Based recognition system:
In the Fingerprint based existing attendance system, a portable fingerprint device needs to be configured with the students fingerprint earlier. Later either during the lecture hours or before, the student needs to record the fingerprint on the configured device to

ensure their attendance for the day. The problem with this approach is that during the lecture time it may distract the attention of the students.

2. RFID (Radio Frequency Identification) Based recognition system:

In the RFID based existing system, the student needs to carry a Radio Frequency Identity Card with them and place the ID on the card reader to record their presence for the day. The system is capable of to connect to RS232 and record the attendance to the saved database. There are possibilities for the fraudulent access may occur. Some are students may make use of other student’s ID to ensure their presence when the student is absent, or they even try to misuse it sometimes.

3. Iris Based Recognition System:

In the Iris based student attendance system, the student needs to stand in front of a camera, so that the camera will scan the Iris of the student. The scanned iris is matched with data of student stored in the database and the attendance on their presence needs be updated. This reduces the paper and pen workload of the faculty member of the institute. This also reduces the chances of proxies in the class and helps in maintaining the student records safe. It is a wireless biometric technique that solves the problem of spurious attendance and the trouble of laying the corresponding network.

4. Face Based Recognition System:

The facial recognition technology can be used in recording the attendance through a high-resolution digital camera that detects and recognizes the faces of the students and the machine compares the recognized face with students’ face images stored in the database. Once the face of the student is matched with the stored image, then the attendance is marked in attendance database for further calculation. If the captured image doesn't match with the students' face present in the database, then this image is stored as a new image onto the database. In this system, there are possibilities for the camera to not to capture the image properly or it may miss some of the students from capturing.

1.7 Drawbacks in existing system

¬ These attendance systems are manual

¬ There is always a chance of forgery (one person signing the presence of the other one) Since these are manual so there is great risk of error

¬ More manpower is required (some person to take attendance)

¬ Calculations related to attendance are done manually (total classes attended in month) which is prone to error. It is difficult to maintain database or register in manual systems

¬ It is more costly (price of register, pen and the salary of person taking attendance)

It is difficult to search for data from this system (especially if that data we are asking for is of every long ago)

The previous approach in which manually takes and maintains the attendance records was very inconvenient task. Traditionally, student’s attendances are taken manually by using attendance sheet given by the faculty members in class, which is a time-consuming event. Moreover, it is very difficult to verify one by one student in a large classroom environment with distributed branches whether the authenticated students are responding or not.

The ability to compute the attendance percentage becomes a major task as manual computation produces errors and wastes a lot of time. This method could easily allow for impersonation and the attendance sheet could be stolen or lost.

CHAPTER 2

LITERATURE SURVEY

Face detection is a computer technology that determines the location and size of human face in arbitrary (digital) image. The facial features are detected and any other objects like trees, buildings and bodies etc are ignored from the digital image. It can be regarded as a _specific ‘case of object-class detection, where the task is finding the location and sizes of all objects in an image that belong to a given class. Face detection can be regarded as a more _general ‘case of face localization. In face localization, the task is to find the locations and sizes of a known number of faces (usually one). Basically, there are two types of approaches to detect facial part in the given image i.e. feature base and image base approach. Feature base approach tries to extract features of the image and match it against the knowledge of the face features. While image base approach tries to get best match between training and testing images.

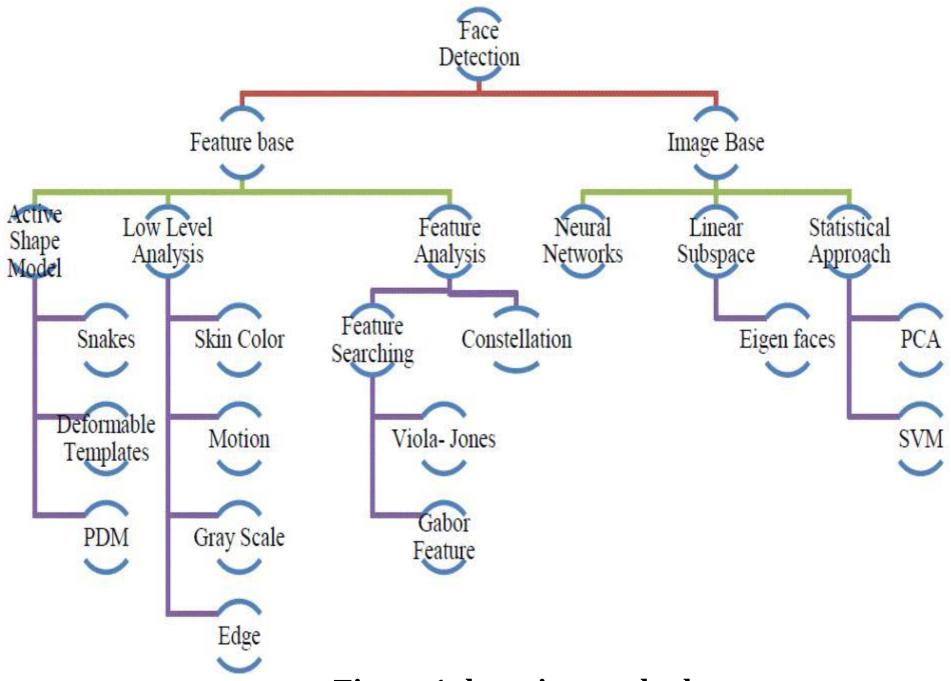


Figure 1 detection methods

2.1 FEATURE BASE APPROCH:

Active Shape Model Active shape models focus on complex non-rigid features like actual physical and higher-level appearance of features Means that Active Shape

Models (ASMs) are aimed at automatically locating landmark points that define the shape of any statistically modelled object in an image. When of facial features such as the eyes, lips, nose, mouth and eyebrows. The training stage of an ASM involves the building of a statistical

a) facial model from a training set containing images with manually annotated landmarks. ASMs is classified into three groups i.e. snakes, PDM, Deformable templates

b) 1.1) Snakes: The first type uses a generic active contour called snakes, first introduced by Kass et al. in 1987 Snakes are used to identify head boundaries [8,9,10,11,12]. In order to achieve the task, a snake is first initialized at the proximity around a head boundary. It then locks onto nearby edges and subsequently assume the shape of the head. The evolution of a snake is achieved by minimizing an energy function, Esnake (analogy with physical systems), denoted asEsnake = Einternal + EExternal WhereEinternal and EExternal are internal and external energy functions. Internal energy is the part that depends on the intrinsic properties of the snake and defines its natural evolution. The typical natural evolution in snakes is shrinking or expanding. The external energy counteracts the internal energy and enables the contours to deviate from the natural evolution and eventually assume the shape of nearby features—the head boundary at a state of equilibria. Two main consideration for forming snakes i.e. selection of energy terms and energy minimization. Elastic energy is used commonly as internal energy. Internal energy is varying with the distance between control points on the snake, through which we get contour an elastic-band characteristic that causes it to shrink or expand. On other side external energy relay on image features. Energy minimization process is done by optimization techniques such as the steepest gradient descent. Which needs highest computations. Huang and Chen and Lam and Yan both employ fast iteration methods by greedy algorithms. Snakes have some demerits like contour often becomes trapped onto false image features and another one is that snakes are not suitable in extracting nonconvex features.

2.1.1 Deformable Templates:

Deformable templates were then introduced by Yuille et al. to consider the a priori of facial features and to better the performance of snakes. Locating a facial feature boundary is not an easy task because the local evidence of facial edges is difficult to organize into a sensible global entity using generic contours. The low brightness contrast around some of these features also makes the edge detection process. Yuille et al. took the concept of snakes a step further by incorporating global information of the eye to improve the reliability of the extraction process. Deformable templates approaches are developed to solve this problem. Deformation is based on local valley, edge, peak, and brightness. Other than face boundary, salient feature (eyes, nose, mouth and eyebrows) extraction is a great challenge of face recognition. E = Ev + Ee

+ Ep + Ei + Einternal; where Ev, Ee, Ep, Ei, Einternal are external energy due to valley, edges, peak and image brightness and internal energy

2.1.2 PDM (Point Distribution Model):

Independently of computerized image analysis, and before ASMs were developed, researchers developed statistical models of shape. The idea is that once you represent shapes as vectors, you can apply standard statistical methods to them just like any other multivariate object. These models learn allowable constellations of shape points from training example sand use principal components to build what is called a Point Distribution Model. These have been used in diverse ways, for example for categorizing Iron Age broaches. Ideal Point Distribution Models can only deform in ways that are characteristic of the object. Cootes and his colleagues were seeking models which do exactly that so if a beard, say, covers the chin, the shape model can \override the image" to approximate the position of the chin under the beard. It was therefore natural (but perhaps only in retrospect) to adopt Point Distribution Models. This synthesis of ideas from image processing and statistical shape modelling led to the Active Shape Model. The first parametric statistical shape model for image analysis based on principal components of inter-landmark distances was presented by Cootes and Taylor in. On this approach, Cootes, Taylor, and their colleagues, then released a series of papers that cumulated in what we call the classical Active Shape Model.

2.2) LOW LEVEL ANALYSIS:

Based on low level visual features like color, intensity, edges, motion etc. Skin Color BaseColor is avital feature of human faces. Using skin-color as a feature for tracking a face has several advantages. Color processing is much faster than processing other facial features. Under certain lighting conditions, color is orientation invariant. This property makes motion estimation much easier because only a translation model is needed for motion estimation. Tracking human faces using color as a feature has several problems like the color representation of a face obtained by a camera is influenced by many factors (ambient light, object movement, etc.

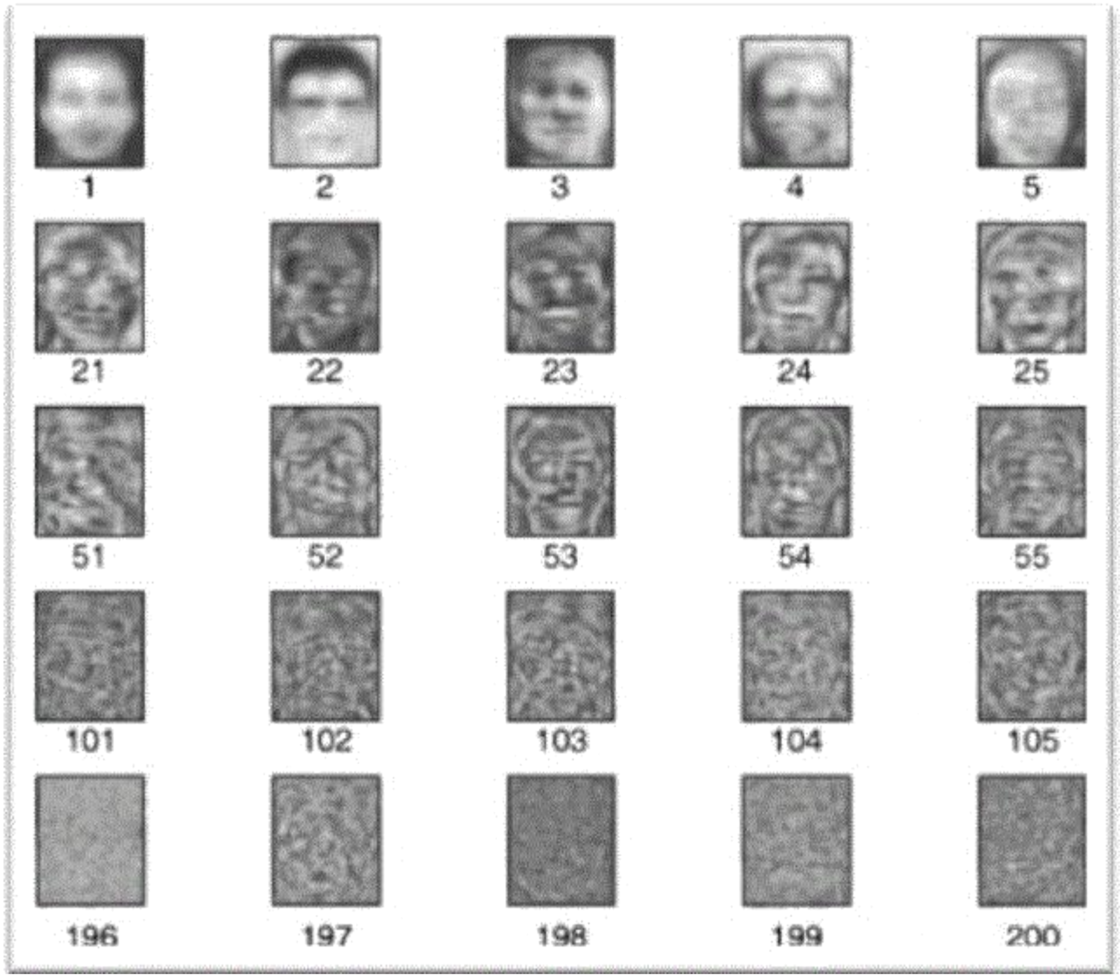


Figure 2. face detection

Majorly three different face detection algorithms are available based on RGB, YCbCr, and HIS color space models. In the implementation of the algorithms there are three main steps viz.

(1) Classify the skin region in the color space,

(2) Apply threshold to mask the skin region and

(3) Draw bounding box to extract the face image.

Crowley and Coutaz suggested simplest skin color algorithms for detecting skin pixels.

The perceived human color varies as a function of the relative direction to the illumination.

The pixels for skin region can be detected using a normalized color histogram and can be normalized for changes in intensity on dividing by luminance. Converted an [R, G, B] vector is converted into an [r, g] vector of normalized color which provides a fast means of skin detection. This algorithm fails when there is some more skin region like legs, arms, etc. Cahi and Ngan [27] suggested skin color classification algorithm with YCbCr color space. Research found that pixels belonging to skin region having similar Cb and Cr values. So that the thresholds be chosen as [Cr1, Cr2] and [Cb1, Cb2], a pixel is classified to have skin tone if the values [Cr, Cb] fall within the thresholds. The skin color distribution gives the face portion in the color image. This algorithm is also having the constraint that the image should be having only face as the skin region. Kjeldson and Kender defined a color predicatein HSV color space to separate skin regions from background. Skin color classification in HSI color space is the same as YCbCr color space but here the responsible values are hue (H) and saturation (S). Like above the threshold be chosen as [H1, S1] and [H2, S2], and a pixel is classified to have skin tone if the values [H, S] fall within the threshold and this distribution gives the localized face image. Like above two algorithm this algorithm is also having the same constraint.

2.3) MOTION BASE:

When use of video sequence is available, motion information can be used to locate moving objects. Moving silhouettes like face and body parts can be extracted by simply thresholding accumulated frame differences. Besides face regions, facial features can be located by frame differences.

2.3.1 Gray Scale Base:

Gray information within a face can also be treat as important features. Facial features such as eyebrows, pupils, and lips appear generally darker than their surrounding facial regions. Various recent feature extraction algorithms search for local gray minima within segmented facial regions. In these algorithms, the input images are first enhanced by contrast-stretching and gray-scale morphological routines to improve the quality of local dark patches and thereby make detection easier. The extraction of dark patches is achieved by low-level gray-scale thresholding. Based method and consist three levels. Yang and huang presented new approach i.e. face gray scale behaviour in pyramid (mosaic) images. This system utilizes hierarchical Face location consist three levels. Higher two level based on mosaic images at different resolution. In the lower level, edge detection method is proposed. Moreover, this algorithm gives fine response in complex background where size of the face is unknown.

2.3.2 Edge Base:

Face detection based on edges was introduced by Sakai et al. This work was based on analysing line drawings of the faces from photographs, aiming to locate facial features. Then later Craw et al. proposed a hierarchical framework based on Sakai et al. ‘work to trace a human head outline. Then after remarkable works were carried out by many researchers in this specific area. Method suggested by Anila and Devarajan was very simple and fast. They proposed framework which consist three stepsi.e. initially the images are enhanced by applying median filter for noise removal and histogram equalization for contrast adjustment. In the second step the edge image is constructed from the enhanced image by applying sobel operator. Then a novel edge tracking algorithm is applied to extract the sub windows from the enhanced image based on edges. Further they used Back propagation Neural Network (BPN) algorithm to classify the sub-window as either face or non-face.

2.4 FEATURE ANALYSIS

These algorithms aim to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. These methods are designed mainly for face localization

2.4.1 Feature Searching

Viola Jones Method:

Paul Viola and Michael Jones presented an approach for object detection which minimizes computation time while achieving high detection accuracy. Paul Viola and Michael Jones [39] proposed a fast and robust method for face detection which is 15 times quicker than any technique at the time of release with 95% accuracy at around 17 fps.The technique relies on the use of simple Haar-like features that are evaluated quickly through the use of a new image representation. Based on the concept of an ―Integral Image‖ it generates a large set of features and uses the boosting algorithm AdaBoost to reduce the overcomplete set and the introduction of a degenerative tree of the boosted classifiers provides for robust and fast interferences. The detector is applied in a scanning fashion and used on gray-scale images, the scanned window that is applied can also be scaled, as well as the features evaluated.

Gabor Feature Method:

Sharif et al proposed an Elastic Bunch Graph Map (EBGM) algorithm that successfully implements face detection using Gabor filters. The proposed system applies 40 different Gabor filters on an image. As a result of which 40 images with different angles and orientation are received. Next, maximum intensity points in each filtered image are calculated and mark them as fiducial points. The system reduces these points in accordance to distance between them. The next step is calculating the distances between the reduced points using distance formula. At last, the distances are compared with database. If match occurs, it means that the faces in the image are detected. Equation of Gabor filter is shown below`

$$\psi_{u,v}(z) = \frac{\| k_{u,v} \|^2}{\sigma^2} e^{\left(-\frac{\|k_{u,v}\|^2 \|z\|^2}{2\sigma^2} \right)} \left[e^{i\vec{k}_{u,v}z} - e^{-\frac{\sigma^2}{2}} \right]$$

Where

$\phi_u = \frac{u\pi}{8}, \quad \phi_u \in [0, \pi)$ gives the frequency,

2.5 CONSTELLATION METHOD

All methods discussed so far can track faces but still some issue like locating faces of various poses in complex background is truly difficult. To reduce this difficulty investigator, form a group of facial features in face-like constellations using more robust modelling approaches such as statistical analysis. Various types of face constellations have been proposed by Burl et al. They establish use of statistical shape theory on the features detected from a multiscale Gaussian derivative filter. Huang et al. also apply a Gaussian filter for pre-processing in a framework based on image feature analysis. Image Base Approach.

2.5.1 Neural Network

Neural networks gaining much more attention in many pattern recognition problems, such as OCR, object recognition, and autonomous robot driving. Since face detection can be treated as a two-class pattern recognition problem, various neural network algorithms have been proposed. The advantage of using neural networks for face detection is the feasibility of training a system to capture the complex class conditional density of face patterns. However, one demerit is that the network architecture must be extensively tuned (number of layers, number of nodes, learning rates, etc.) to get exceptional performance. In early days most hierarchical neural network was proposed by Agui et al. [43]. The first stage having two parallel subnetworks in which the inputs are filtered intensity values from an original image. The inputs to the second stage network consist of the outputs from the sub networks and extracted feature values. An output at the second stage shows the presence of a face in the input region. Propp and Samal developed one of the earliest neural networks for face detection [44]. Their network consists off-our layers with 1,024 input units, 256 units in the first hidden layer, eight units in the second hidden layer, and two output units. Feraud and Bernier presented a detection method using auto associative neural networks [45], [46], [47]. The idea is based on [48] which shows an auto associative network with five layers can perform a nonlinear principal component analysis. One auto associative network is used to detect frontal-view faces and another one is used to detect faces turned up to 60 degrees to the left and right of the frontal view. After that Lin et al. presented a face detection system using probabilistic decision-based neural network (PDBNN) [49]. The architecture of PDBNN is like a radial basis function (RBF) network with modified learning rules and probabilistic interpretation.

2.6 LINEAR SUB SPACE METHOD

Eigen faces Method:

An early example of employing eigen vectors in face recognition was done by Kohonen in which a simple neural network is demonstrated to perform face recognition for aligned and normalized face images. Kirby and Sirovich suggested that images of faces can be linearly encoded using a modest number of basis images. The idea is arguably proposed first by Pearson in 1901 and then by HOTELLING in 1933 .Given a collection of n by m pixel training.

Images represented as a vector of size m X n, basis vectors spanning an optimal subspace are determined such that the mean square error between the projection of the training images onto this subspace and the original images is minimized. They call the set of optimal basis vectors Eigen pictures since these are simply the eigen vectors of the covariance matrix computed from the vectorized face images in the training set. Experiments with a set of 100 images show that a face image of 91 X 50 pixels can be effectively encoded using only50 Eigen pictures.

2.7 STATISTICAL APPROCH

Support Vector Machine (SVM):

SVMs were first introduced Osuna et al. for face detection. SVMs work as a new paradigm to train polynomial function, neural networks, or radial basis function (RBF) classifiers. SVMs works on induction principle, called structural risk minimization, which targets to minimize an upper bound on the expected generalization error. An SVM classifier is a linear classifier where the separating hyper plane is chosen to minimize the expected classification error of the unseen test patterns. In Osunaet al. developed an efficient method to train an SVM for large scale problems and applied it to face detection. Based on two test sets of 10,000,000 test patterns of 19 X 19 pixels, their system has slightly lower error rates and runs approximately30 times faster than the system by Sung and Poggio. SVMs have also been used to detect faces and pedestrians in the wavelet domain.

CHAPTER 3

PROPOSED SOLUTION

To overcome the problems in existing attendance system we shall develop a Biometric based attendance system over simple attendance system. Interactive system over static one. Digitized attendance system over file system. There are many solutions to automate the attendance management system like thumb-based system, simple computerized attendance system but all these systems have limitations over work and security point of view. Our proposed system shall be a "Face Recognition Attendance System" which uses the basic idea of image processing, which is used in many secure applications like banks, airports etc.

3.1 Proposed System Components

Following are the main components of the proposed system

1. Student Registration
2. Face Detection
3. Face Recognition
   - Feature Extraction
   - Feature Classification
4. Attendance management system

It will allow uploading, updating and deletion of the contents of the system.
Attendance management will handle:

- Automated Attendance marking
- Manual Attendance marking
- Attendance details of users.

3.2 Proposed System Outcome

¬ System will only allow authenticated user to login to the system and/or make changes to it.

¬ It will allow user to mark attendance of the students via face recognition technique.

¬ It will detect faces via webcam and then recognize the faces.

¬ After recognition it will mark the attendance of the recognized student and update the attendance record.

¬ The user will be able to print these record details afterward.

CHAPTER 4

What contribution would the project make?

Face recognition is the most natural biological features recognition technology according to the cognitive rule of human beings; its algorithm is ten times more complex than a fingerprint algorithm.

Face recognition is featured by the following advantages compared to fingerprint:

1. Accurate and Fast Identification

Industrial Leading Facial Recognition Algorithm, match more data than fingerprint, FAR<0.0001%

2. High Usability and Security

Failure to enrol and acquire rate is less than 0.0001%, fingerprint technology will have problems for enrolment with cold, wet, desquamation, elder, and around 5% people cannot get enrolled with fingerprint technology

Incident track able for security with photo which captured by camera, there is no evidence with fingerprint technology to track the incident.

3. User friendly design

Contact less authentication for ultimate hygiene.

CHAPTER 5
PROJECT CHARTER

5.1 Project Definition

Face Recognition is a biometric method of identifying an individual by comparing live capture or digital image data with the stored record for that person.

Face Recognition Attendance System is marking of attendance based on this technology.

5.2 Project Scope

- Provides an automated attendance system that is practical, reliable and eliminate disturbance and time loss of traditional attendance systems.
- Present a system that can accurately evaluate student’s performance depending on their recorded attendance rate.

5.3 Team Size

Our project team consist of two students namely:

1. Salman Husain.
2. Vishwa Mohan

CHAPTER 6

PROJECT PLAN

6.1 Objectives

¬ Detection of unique face image amidst the other natural component such as walls and other backgrounds.

¬ Detection of faces amongst other face characters such as beard, spectacles etc.

¬ Extraction of unique characteristic features of a face useful for face recognition.

¬ Effective recognition of unique faces in a class (individual recognition).

¬ Automated update in the attendance sheet without human intervention.

6.2 Goals

¬ To help the lecturers, improve and organize the process of track and manage student attendance.

¬ Provides a valuable attendance service for both teachers and students.

¬ Reduce manual process errors by provide automated and a reliable attendance system.

¬ Increase privacy and security which student cannot present him or his friend while they are not.

¬ Produce monthly reports for lecturers.

¬ Flexibility, Lectures capability of editing attendance records.

6.3 Software Model - Waterfall Model

Waterfall Model is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order.

In the waterfall model, you must plan and schedule all the activities before starting, working on them (plan-driven process).

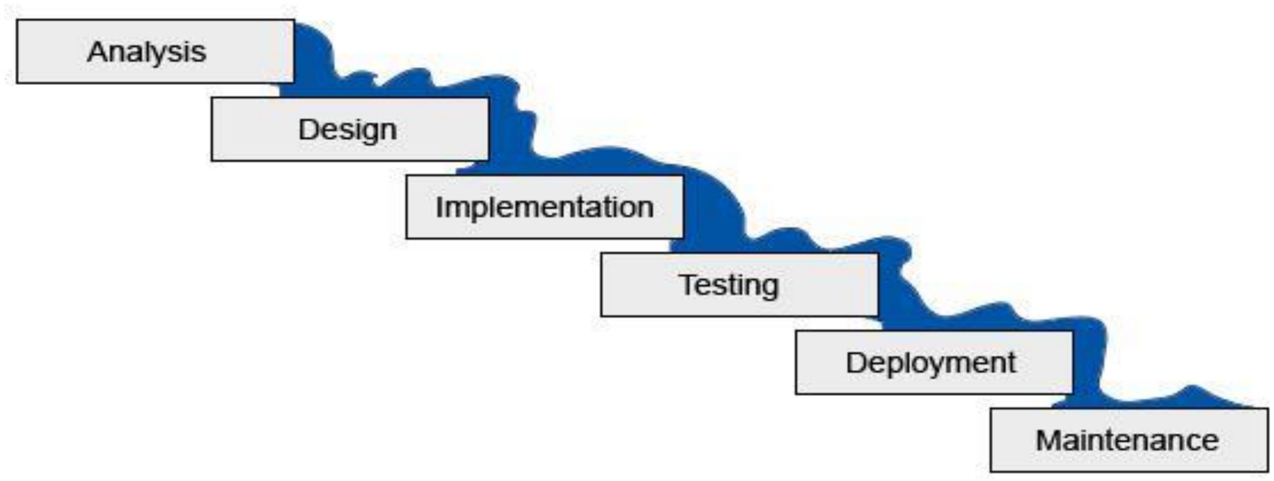


Figure 3 Waterfall Model

6.4 Strategies

a) **Finalize Project Details**

- Find information regarding our project.
- Gathering more detail about what we required to learn.
- What are our expectations from project?
- Listing the scope of the project.
- Develop a detailed plan and define goals.
- Creating measurable criteria for success.
- Listing roles and responsibilities of team members

b) **Set Clear Expectations**

- Checking that we have everything which can lead us to a successful project.
- Being clear about which team members are responsible for all the components of a project.

- Work contribution makes accountability clear, what member should perform which task.
- Conveying information and explaining what we are going to do on project to the teacher in the project proposal.

c) **Choose the Right Team and System**

- First, we choose the team members as our plans and expectation were clear.
- Distributing work according skills, talent and personalities that are right for each project module.
- Tried to avoid having too many people on a team as well, so the communication goes easy.

d) **Define Milestones**

- It is important to define key milestones throughout the lifecycle of the project.
- We started by including four main phases: initiation, planning, execution, and closure.
- Performing an evaluation after each phase to know how our team is doing by examining deliverables.
- This process keeps you informed about any problems that arise while ensuring that each phase of the project is completed successfully.

e) Establish Clear Communication

- The element that can make or break a project is communication, we have focused on it and tried to make it fluid.
- Time to time sharing the progress report to each other.
- Sharing of any major issues and discussing it together.

f) Manage Project Risks

- There is risk in every project, so evaluating it at the start and manage it.
- Keeping backup plans ready for preventing stuck situation in project.
- Completing modules of decided time so we get enough time to test every functionality perfectly and if any problem arises, we get enough time to solve it.

g) Avoid Scope Creep

- Its duty of everyone to keep project on track.
- It is important to know how much change can occur before affecting deadlines and deliverables.
- Scope creep generally takes place when there are additions to a project, which is not revised accordingly.

h) Evaluate the Project after Completion

- Each project provides information that you can utilize in the future.
- Therefore, reviewing the project is such a valuable practice.
- When project members know what went right, what went wrong, and how to adjust next time, they can develop best practices for future work.

6.5 Work Division (w.r.t team members)

Vishwa Mohan

- Graphical User Interface
- Registration
- Face Recognition
- Generating attendance in Excel Sheet

Salman Husain

- ¬ Face Detection
- ¬ Database Storage
- ¬ Face Recognition
- ¬ Registered Students in Excel sheet

6.6 Tools and Technologies

Tools

- ¬ MATLAB
- ¬ Microsoft Excel
- ¬ www.lucidchart.com
- ¬ www.teamgantt.com

Technologies

MATLAB

- Computer vision toolbox
- Image acquisition toolbox
- Image processing toolbox
- Parallel Computing toolbox

CHAPTER 7
SYSTEM REQUIREMENT SPECIFICATION

I. Technical Requirement

i. Hardware Requirements

¬ A standalone computer

¬ Web Camera must be used to obtain the snapshots.

¬ Secondary memory to store all the images and database

ii. Software requirements

¬ MATLAB Version 8.5.0(R2015a) or higher

¬ Windows 8 or higher

¬ Open Source Computer Vision.

II. Functional Requirements

System functional requirement describes activities and services that must provide.

¬ User must be able to manage student records.

¬ Only authorized user must be able to use the system.

¬ System must be attached to webcam and face recognition should be smooth.

¬ The administrator or the person who will be given the access to the system must login into system before using it.

¬ The information must be entered and managed properly.

III. Non-Functional Requirements

Non-functional Requirements are characteristics or attributes of the system that can judge its operation. The following points clarify them:

a. Accuracy and Precision: the system should perform its process in accuracy and precision to avoid problems.

b. Flexibility: the system should be easy to modify, any wrong should be correct.

c. Security: the system should be secure and saving student's privacy.

d. Usability: the system should be easy to deal with and simple to understand.

e. Maintainability: the maintenance group should be able to cope up with any problem when occurs suddenly.

f. Speed and Responsiveness: Execution of operations should be fast.

Non–Functional Requirements are as follow:

¬ The GUI of the system will be user friendly.

¬ The data that will be showed to the users will be made sure that it is correct and is available for the time being.

¬ The system will be flexible to changes.

¬ The system will be extensible for changes and to the latest technologies.

¬ Efficiency and effectiveness of the system will be made sure.

¬ The performance of the system will be made sure.

IV. Student Requirements

¬ Student needs to enter the proper details while registering him/her.

¬ He/ She needs to sit properly and capture 10-15 images of himself/herself in different direction and expressions.

¬ At the time of taking attendance, students need to sit properly facing the camera.

V. Teaching Staff Requirements

¬ The faculty needs to login the system at the time of attendance.

¬ The faculty needs to enter lecture details before starting the attendance process.

¬ If the entered lecture details don’t match with the ones in the database (excel sheet) an error dialog will be displayed.

¬ As the students are recognized by the system, the attendance report will be generated and shown to the faculty.

VI. Administrator Requirements

¬ The administrator needs to login the system at the time of registering the students for the face recognition process.

¬ He / She must make sure that the student enters the details properly.

¬ Only the administrator has the rights to manage any changes in the system.

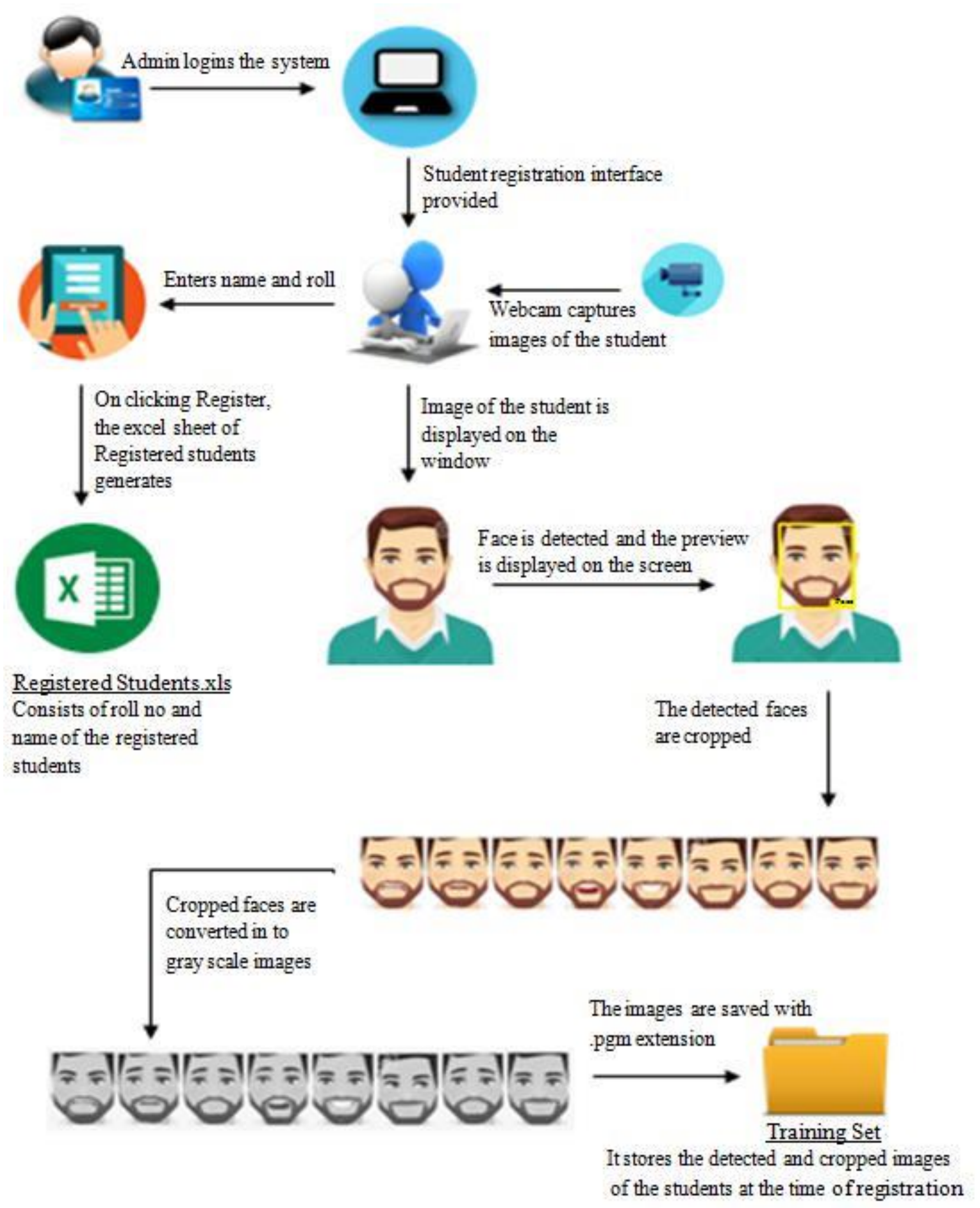
- Only the administrator can view the Training set and the Testing set.
- Only the administrator has the rights to manage any changes in the stored data set.

CHAPTER 8

SYSTEM ANALYSIS

8.1 System Flow

8.1.1 Student Registration Flow



Admin logins the system
Student registration interface provided
Enters name and roll
Webcam captures images of the student
On clicking Register, the excel sheet of Registered students generates
Image of the student is displayed on the window
Face is detected and the preview is displayed on the screen
Registered Students.xls
Consists of roll no and name of the registered students
The detected faces are cropped
Cropped faces are converted in to gray scale images
The images are saved with .pgm extension
Training Set
It stores the detected and cropped images of the students at the time of registration

Figure 4 Student Registration Flow

8.1.2 Taking Attendance Flow

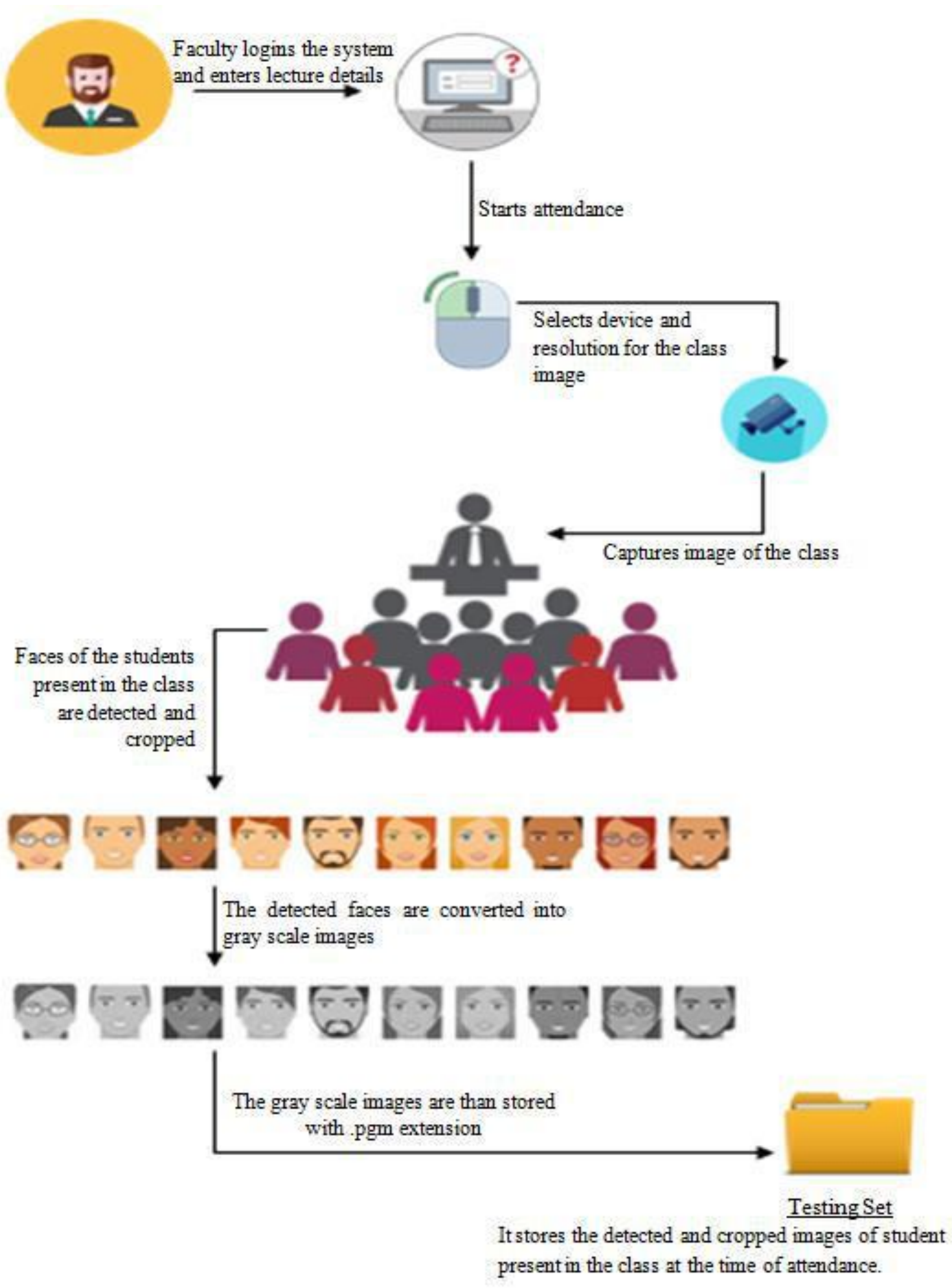

Faculty logins the system and enters lecture details
Starts attendance
Selects device and resolution for the class image
Captures image of the class
Faces of the students present in the class are detected and cropped
The detected faces are converted into gray scale images
The gray scale images are than stored with .pgm extension
Testing Set
It stores the detected and cropped images of student present in the class at the time of attendance.

Figure 5 Taking Attendance Flow

Feature Extraction

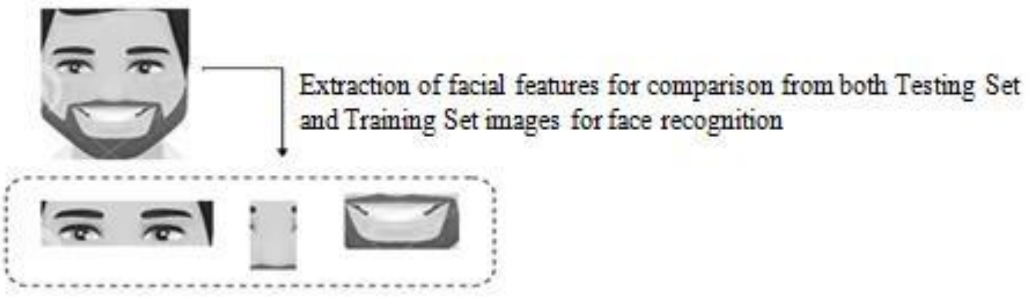

Extraction of facial features for comparison from both Testing Set and Training Set images for face recognition

Figure 6 Feature Extraction

8.1.3 Generating Attendance Flow

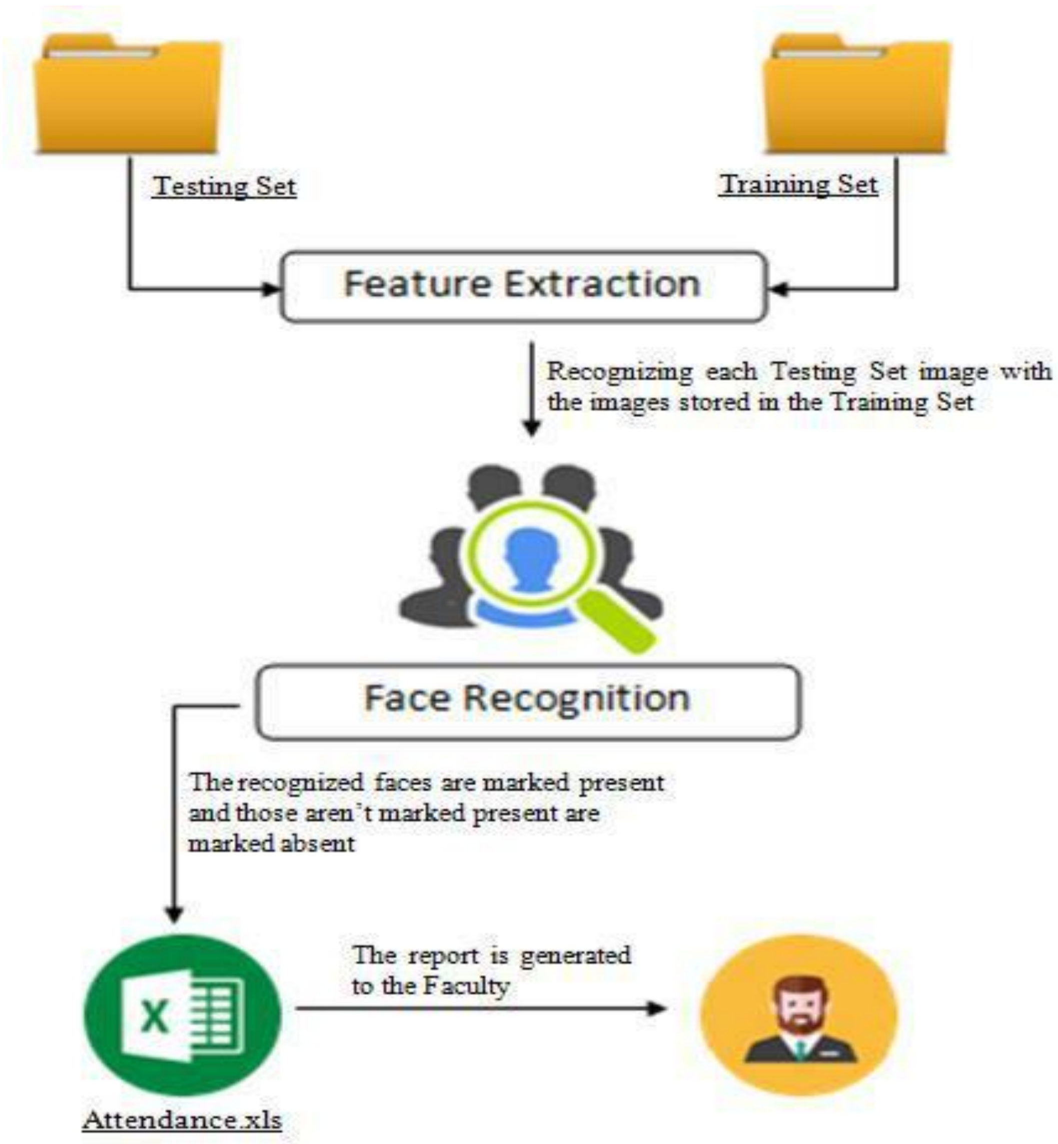

Testing Set
Training Set
Feature Extraction
Recognizing each Testing Set image with the images stored in the Training Set
Face Recognition
The recognized faces are marked present and those aren't marked present are marked absent
The report is generated to the Faculty
Attendance.xls

Figure 7 Generating Attendance Flow

8.2 System Architecture

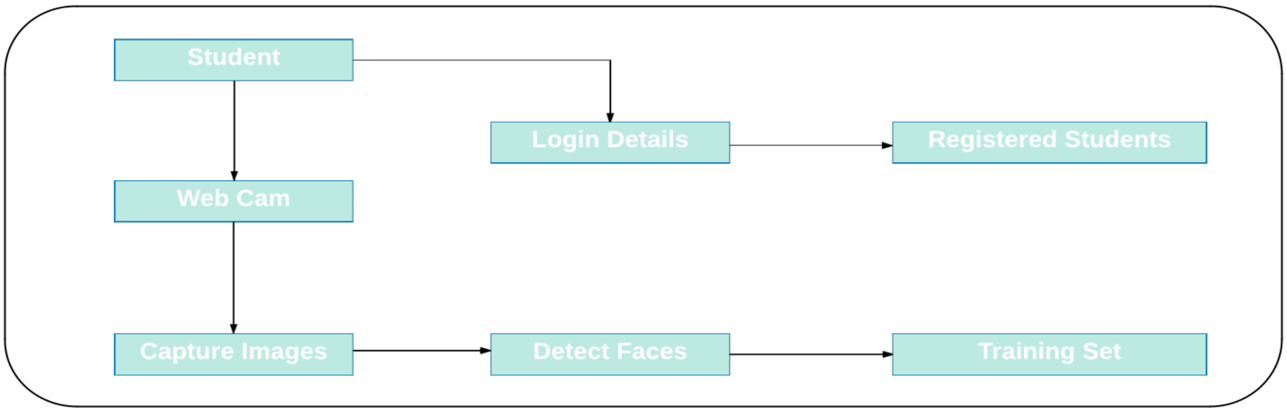


Figure 8 System Architecture (Training Phase: Registration)

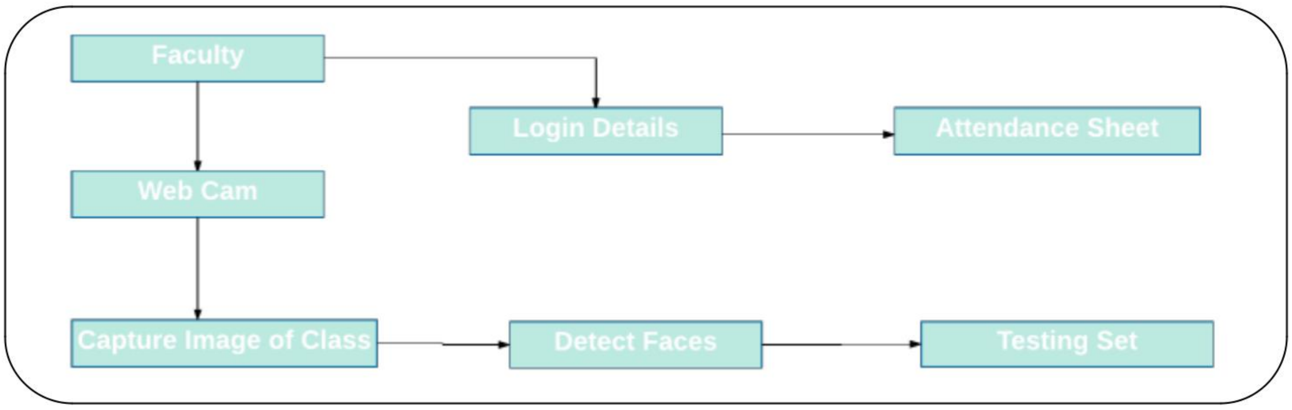


Figure 9 System Architecture (Testing Phase: Attendance)

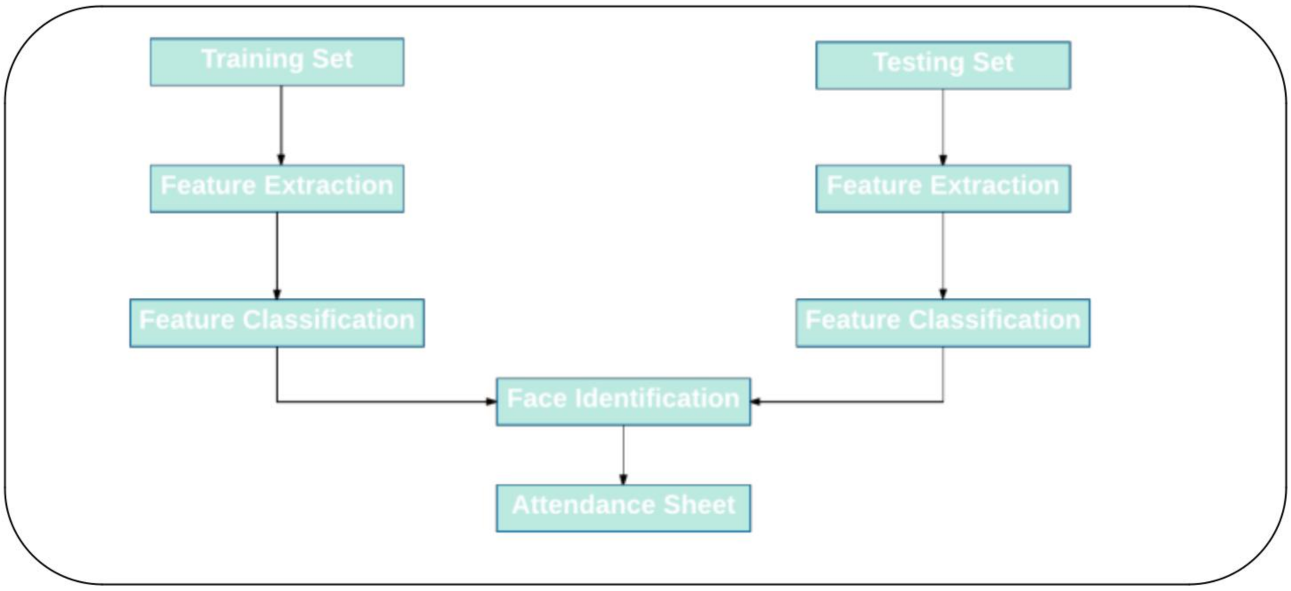


Figure 10 System Architecture (Recognition Phase)

8.3 System Description

The system consists of a camera that captures the images of the classroom and sends it to the image enhancement module. After enhancement the image comes in the Face Detection and Recognition modules and then the attendance is marked on the database server.

At the time of enrolment, templates of face images of individual students are stored in the Face database known as Training Set. Here all the faces are detected from the input image and the algorithm compares them one by one with the face database (Testing Set). If any face is recognized, the attendance is marked on the server from where administrator can access and use it for different purposes.

Teachers come in the class and just press a button to start the attendance process and the system automatically gets the attendance without even the intensions of students and teacher. In this way a lot of time is saved, and this is highly securing process no one can mark the attendance of other.

8.4 System Algorithm

This section describes the software algorithms for the system.

The algorithm consists of the following steps

- Image acquisition
- Histogram normalization
- Skin classification
- Face detection
- Face recognition
- Attendance.

Image Acquisition

Image is acquired from the camera that is connecting above the board.

Histogram Normalization

Captured image sometimes have brightness or darkness in it which should be removed for good results. First the RGB (colour) image is converted to the gray scale image for enhancement Colour image is converted to gray scale image for increasing contrast.

Histogram normalization is good technique for contrast enhancement in the spatial domain.

Skin Classification

It is used for increasing the efficiency of the face detection algorithm. Viola Jones algorithm is used for detection and its accuracy can be increased if the skin is classified before the scanning procedure of faces. Pixels those are closely related to the skin become white and all other are black. This binary image uses the thresholding of skin colours.

Face Detection

In this stage faces are detected by marking the rectangle on the faces of the student appearing on the image source.

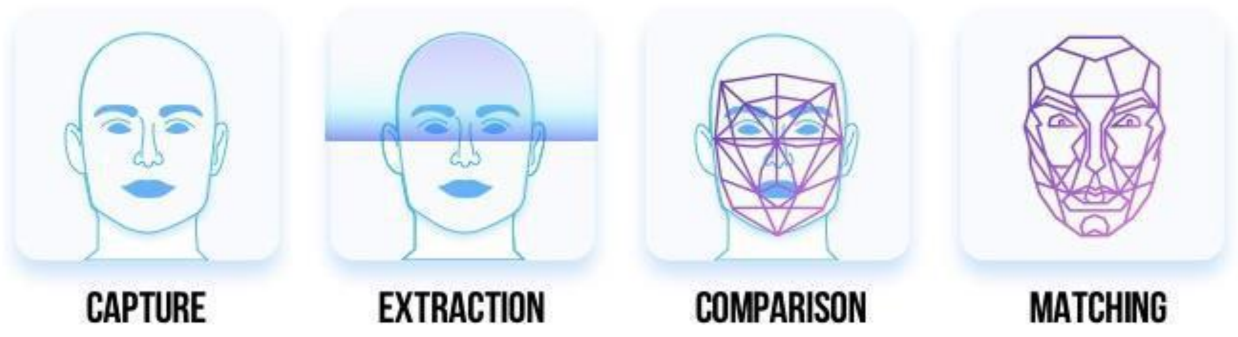
After the detection of faces from the window, the next step is cropping of each detected face.

Each cropped image is assigned as a separate thread for the recognition.

Face and recognition and attendance

After the face detection next step is face recognition. This can be done by cropping the detected faces of students sitting in the class and compare them with the Training Set which stores the multiple images of an individual student at the time of registration.

In this way, the face of each student is verified one by one and attendance is marked in the excel sheet and the report is generated and it is viewed to the faculty.



8.5 Module Specification

8.5.1 Face Detection

It is defined as finding the position of the face of an individual. In other word it can be defined as locating the face region in an image. After detecting the face of human its facial features extracted and has wide range of application like facial expression recognition, face recognition, observation systems, human PC interface and so forth.

For the application of face recognition, detection of face is very important and the first step. Only after detecting face, the face recognition algorithm can be functional. Face detection itself involves some complexities for example surroundings, postures, enlightenment etc.

There are two types of face detection problems:

1) Face detection in images

2) Real-time face detection.

Most face detection systems attempt to extract a fraction of the whole face, thereby eliminating most of the background and other areas of an individual's head such as hair that are not necessary for the face recognition task.

A proper and efficient face detection algorithm always enhances the performance of face recognition systems. Various algorithms are proposed for face detection such as Face geometry-based methods, Feature Invariant methods, Machine learning based methods. Out of all these methods Viola and Jones proposed a framework which gives a high detection rate and is also fast. Viola-Jones detection algorithm is efficient for real time application as it is fast and robust. Hence, we chose Viola-Jones face detection algorithm which makes use of Integral Image and AdaBoost learning algorithm as classifier. We observed that this algorithm gives better results in different lighting conditions and we combined multiple haar classifiers to achieve a better detection rates up to an angle of 30 degrees.

8.5.1.1 Face Detection in Images

With static images, this is often done by running a 'window' across the image. The face detection system then judges if a face is present inside the window. Unfortunately, with static images there is a very large search space of possible locations of a face in an image. Faces may be large or small and be positioned anywhere from the upper left to the lower right of the image. Most face detection systems use an example-based learning approach to decide whether a face is present in the window at that given instant.

A neural network or some other classifier is trained using supervised learning with 'face' and 'non-face' examples, thereby enabling it to classify an image (window in face detection system) as a 'face' or 'non-face'.

There is another technique for determining whether there is a face inside the face detection system's window - using Template Matching. The difference between a fixed target pattern (face) and the window is computed and threshold. If the window contains a pattern which is close to the target pattern (face) then the window is judged as containing a face.

It uses a whole bank of fixed sized templates to detect facial features in an image. By using several templates of different (fixed) sizes, faces of different scales (sizes) are detected. The other implementation of template matching is using a deformable template. Instead of using several fixed size templates, we use a deformable template (which is non-rigid) and thereby change the size of the template hoping to detect a face in an image. A face detection scheme that is related to template matching is image invariants.

Here the fact that the local ordinal structure of brightness distribution of a face remains largely unchanged under different illumination conditions is used to construct a spatial template of the face which closely corresponds to facial features. In other words, the average grey-scale intensities in human faces are used as a basis for face detection. For example, almost always an individual’s eye region is darker than his forehead or nose. Therefore, an image will match the template if it satisfies the 'darker than' and 'brighter than’ relationships.

8.5.1.2 Real-time face detection

It involves detection of a face from a series of frames from a video capturing device. While the hardware requirements for such a system are far more stringent, from a computer vision standpoint, real-time face detection is a far simpler process than detecting a face in a static image. This is because unlike most of our surrounding environment, people are continually moving.

Since in real-time face detection, the system is presented with a series of frames in which to detect a face, by using spatiotemporal filtering (finding the difference between subsequent frames), the area of the frame that has changed can be identified and the individual detected.

Furthermore, exact face locations can be easily identified by using a few simple rules, such as,

1) The head is the small blob above a larger blob -the body

2) Head motion must be reasonably slow and contiguous heads won't jump around erratically.

Real-time face detection has therefore become a relatively simple problem and is possible even in unstructured and uncontrolled environments using this very simple image processing techniques and reasoning rules.

8.5.1.3 Viola Jones for Face Detection

This method has brought together new algorithms and insights to construct a framework for robust and extremely rapid visual detection. This face detection system is most clearly distinguished from previous approaches in its ability to detect faces extremely rapidly. Operating on 384 by 288-pixel images, faces are detected at 15 frames per second on a conventional 700 MHz Intel Pentium III.

In other face detection systems, difference between pixel values of consecutive frames which were present in video or the colour value of each pixel was considered to achieve high frame rates. The information in a grey scale image is worked upon to see if high frame rates have been achieved.

There are four main parts in this method:

1. Integral Images.
2. Haar-like Features.
3. AdaBoost.
4. Cascading Classifier.

Integral Image:

In an Integral Image the value of pixel (x, y) is calculated by adding the value of pixel above and to the left of (x, y).

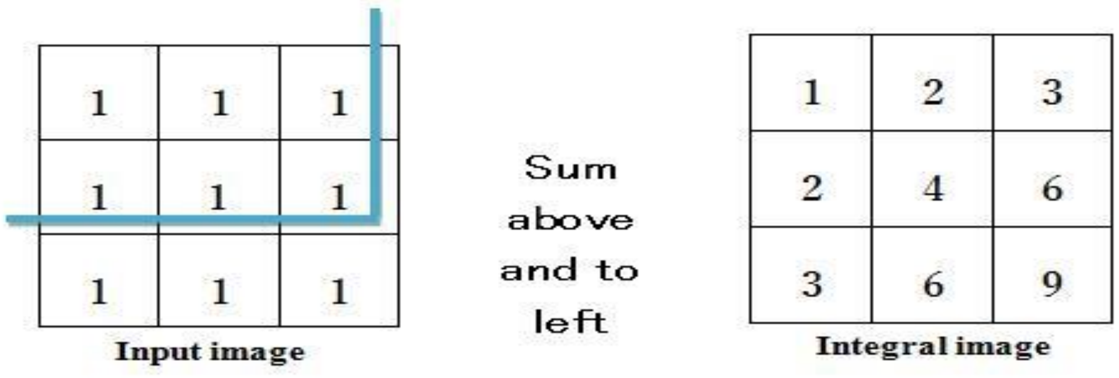


Figure 11 Integral image

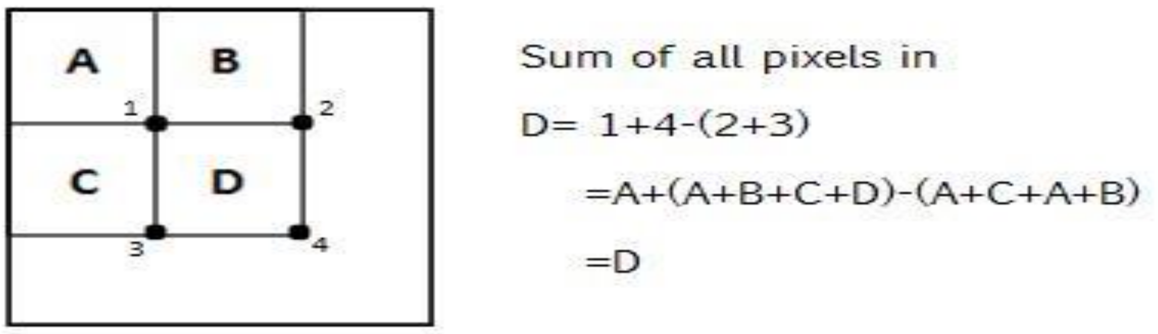


Figure 12 Calculation of Integral image

The calculation of sum of the pixels inside any given rectangle is done using only the corner values of the rectangle.

Haar-Like Features

Haar features are like convolution kernels which are used to detect the presence of that feature in the given image. Feature set is obtained by subtracting sum of all the pixels present below black area from sum of all the pixels present below white area.

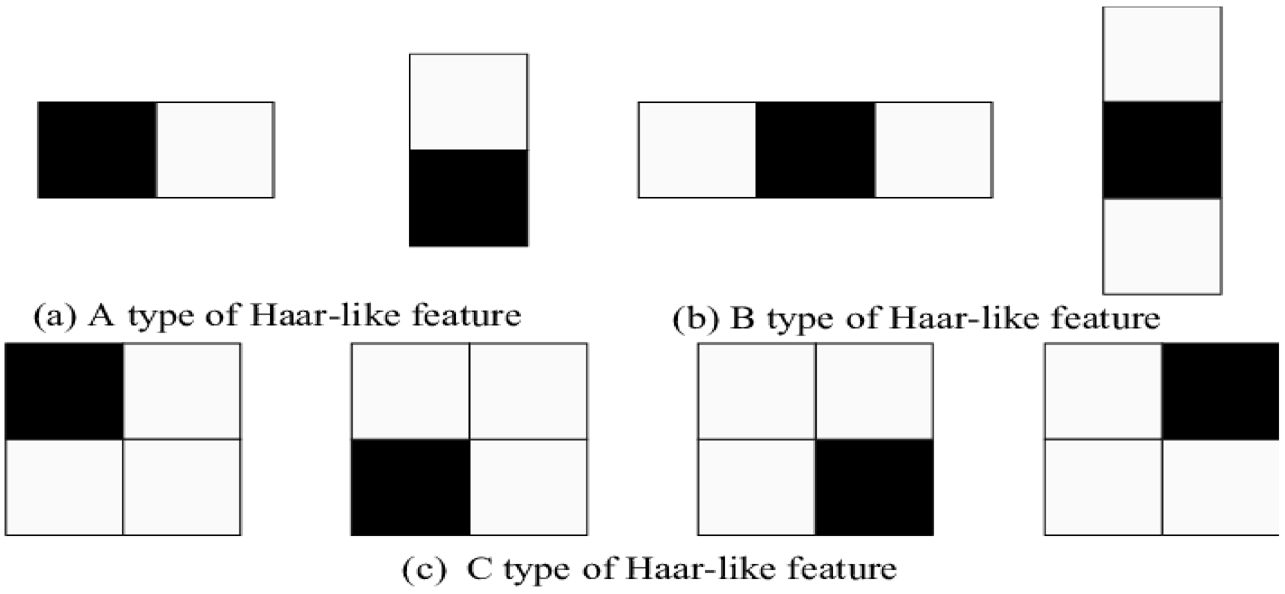

(a) A type of Haar-like feature
(b) B type of Haar-like feature
(c) C type of Haar-like feature

Figure 13 Types of Haar Features

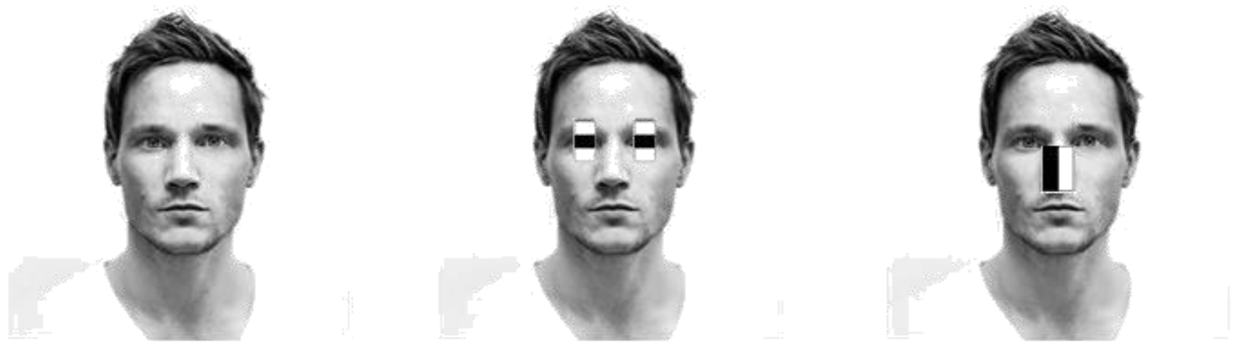


Figure 14 Haar features applied on an image

AdaBoost

Viola Jones algorithm uses 24x24 window as the base window size to start evaluating these features in any given image. If we consider all possible parameters of the haar features like position, scale and type we end up calculating about 160,000+ features in this window.

As stated previously there can be approximately 160,000+ feature values within a detector at 24x24 base resolutions which need to be calculated. But it is to be understood that only few sets of features will be useful among all these features to identify a face.

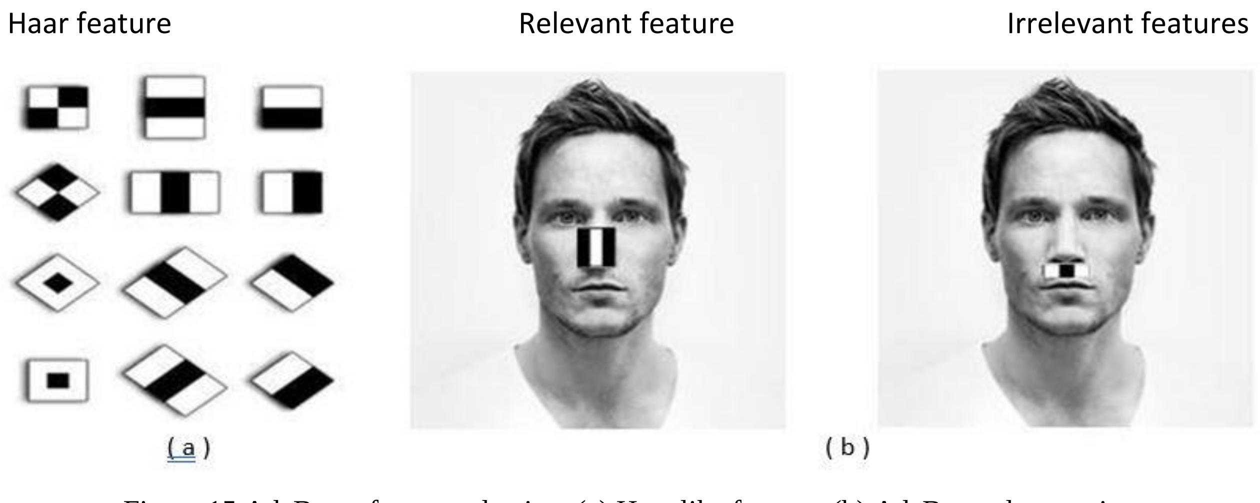


Figure 15 AdaBoost feature selection, (a) Haar like features (b) AdaBoost done on images

In the above image as we can see there is only one Haar feature which can be used to detect nose. So, the other feature which is used in the second image can be ignored as it gives irrelevant feature.

Cascading

The basic principle of the Viola-Jones face detection algorithm is to scan the whole image by consecutively increasing the scanning window size so that the whole image will be covered. Even in an image there is a probability that it may contain one or more faces due to an excessive large amount of the evaluated sub-windows would still be negatives (non-faces).So the algorithm should concentrate on discarding non-faces quickly and spend more on time on probable face regions.

Hence a single strong classifier formed out of linear combination of all best features is not a good to evaluate on each window because of computation cost. A cascade classifier is composed of stages where each stage has certain no of features grouped each containing a strong classifier. The job of each stage is used to determine whether a given sub window is not a face or may be a face. A given sub window is immediately discarded as not a face if it fails in any of the stage.

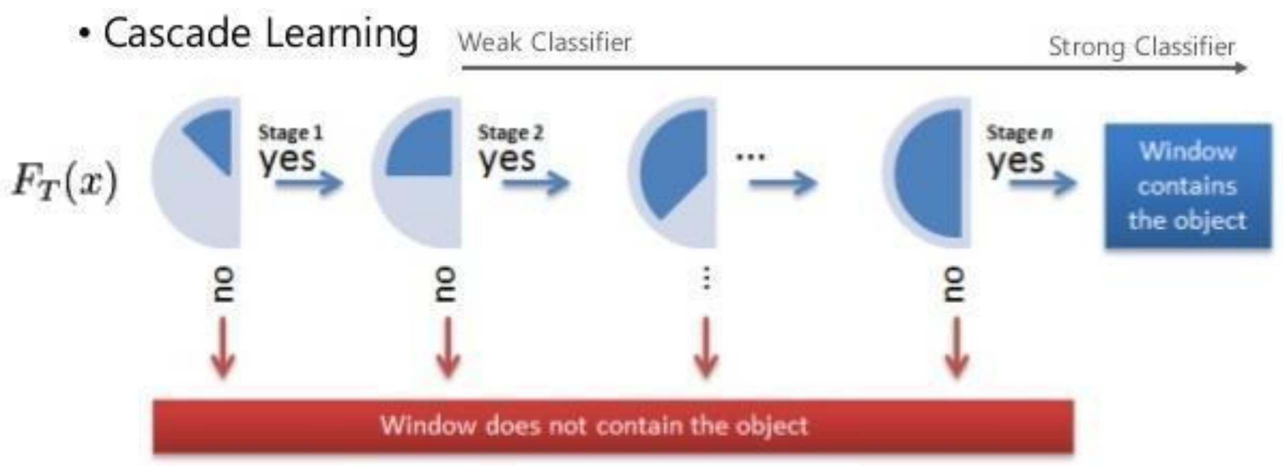


Figure 16 Cascade Classifier

8.5.2 Face Recognition

Face recognition module is used for identifying students in the classrooms then process the related attendance operation.

When images from the face detection module is transferred to database module (Testing Set) and they are ready to be used, face recognition module will use these image and facial information to try to find a match with student images from the database module (Training Set) which stores multiple face images of registered students. If a match is found between image info and database record, then this student is checked as present in the classroom.

There are many face recognition algorithms that are used to recognize a face from an input image.

8.5.2.1 Principal Component Analysis (PCA):

One of the simplest and most effective PCA approaches used in face recognition systems is the so-called eigenface approach. This approach transforms faces into a small set of essential characteristics, eigenface, which are the main components of the initial set of learning images (training set).

Recognition is done by projecting a new image in the eigenface subspace, after which the person is classified by comparing its position in eigenface space with the position of known individuals.

The advantage of this approach over other face recognition systems is in its simplicity, speed and insensitivity to small or gradual changes on the face. The problem is limited to files that can be used to recognize the face. Namely, the images must be vertical frontal views of human faces.

The whole recognition process involves two steps:

1. Feature Extraction process.
2. Recognition process.

Feature Extraction Procedure

1. The first step is to obtain a set S with M face images. In our example M=25 and each image in transformed into a vector of size N and placed into a set.

S= {T1, T2, T3 …. Tm}

2. After you have obtained your set, you will obtain the mean image Ψ.

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n$$

3. Then you will find the difference, Φ between the input image and mean image.

$$\Phi_i = \Gamma_i - \Psi$$

4. In this step, eigenvectors (Eigen face) *ui* and the corresponding eigenvalues should be calculated. The eigenvectors (Eigen face) must be normalized so that they are Eigen vectors, i.e. length of 1. The description of the exact algorithm for determination of eigenvectors and eigenvalues is omitted here, as it belongs to the arsenal of most math programming libraries.

5. We obtain the covariance matrix C in the following manner.

$$C = \frac{1}{M}\sum_{n=1}^{M}\Phi_n\Phi_n^T$$
$$= AA^T$$

$$A = \{\Phi_1, \Phi_2, \Phi_3, \ldots\ldots, \Phi_n\}$$

6. A^T

$$L_{mn} = \Phi_m^T\Phi_n$$

From M eigenvectors (Eigen face) *ui*, only *M'* should be chosen which have the highest eigenvalues. The higher the eigenvalues, the more characteristic features of a face do the particular eigenvector describe. Eigen face with low eigenvalues can be omitted, as they explain only a small part of characteristic features of the face. After *M'* Eigen face *ui* are determined, the training phase of algorithm is finished.

Recognition Procedure

A new face is transformed into its Eigen face components. First, we compare our input image with our mean image and multiply their difference with each eigenvector of the L matrix. Each value would represent a weight and would be saved on a vector Ω.

$$\omega_k = u_k^T (\Gamma - \Psi) \ \Omega^T = [\omega_1, \omega_2, \ \ldots\ldots, \omega_M]$$

We now determine which face class provides the best description for the input image. This is done by minimizing the Euclidean distance.

$$\varepsilon_k = \left\| \Omega - \Omega_k \right\|^2$$

The input face is considered to belong to a class if ε*k* is below an established threshold θε. Then the face image is considered to be a known face. If the difference is above the given threshold but below a second threshold, the image can be determined as an unknown face. If the input image is above these two thresholds, the image is determined not to be a face.

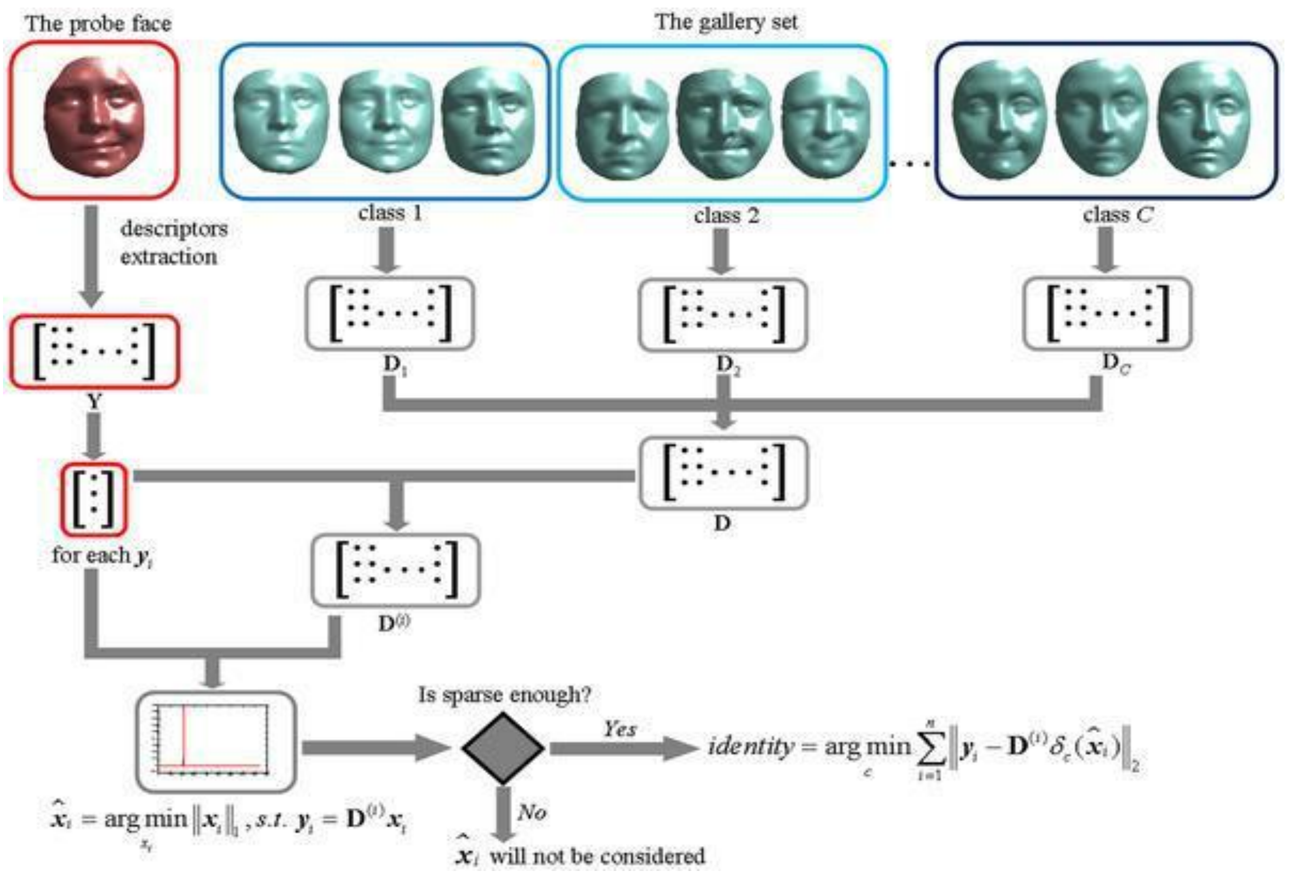

The probe face
The gallery set
class 1
class 2
class C
descriptors extraction
$\mathbf{D}_1$
$\mathbf{D}_2$
$\mathbf{D}_C$
$\mathbf{Y}$
$\mathbf{D}$
for each y_i
$\mathbf{D}^{(i)}$
Is sparse enough?
Yes
No
$identity = \arg\min_c \sum_{i=1}^{n} \| y_i - \mathbf{D}^{(i)} \delta_c(\hat{x}_i) \|_2$
$\hat{x}_i = \arg\min_{x_i} \|x_i\|_1, s.t.\ y_i = \mathbf{D}^{(i)} x_i$
$\hat{x}_i$ will not be considered

Figure 17 Principal Component Analysis

8.5.2.2 Linear Discriminate Analysis (LDA):

Linear Discriminant analysis explicitly attempts to model the difference between the classes of data. LDA is a powerful face recognition technique that overcomes the limitation of Principle component analysis technique by applying the linear discriminant criterion.

This criterion tries to maximize the ratio of the determinant of the between-class scatter matrix of the projected samples to the determinant of the within class scatter matrix of the projected samples. Linear discriminant group images of the same class and separates images of different classes of the images.

Discriminant analysis can be used only for classification not for regression. The target variable may have two or more categories. Images are projected from two dimensional spaces to c dimensional space, where c is the number of classes of the images. To identify an input test image, the projected test image is compared to each projected training image, and the test image is identified as the closest training image.

The LDA method tries to find the subspace that discriminates different face classes. The within- class scatter matrix is also called intrapersonal means variation in appearance of the same individual due to different lighting and face expression. The between-class scatter matrix also called the extra personal represents variation in appearance due to difference in identity.

Linear discriminant methods group images of the same classes and separates images of the different classes. To identify an input test image, the projected test image is compared to each projected training image, and the test image is identified as the closest training image.

8.5.2.3 Linear Binary Pattern Histogram (LBPH):

LBP is really a very powerful method to explain the texture and model of a digital image. A face image is first split into small regions that LBP histograms are extracted and then concatenated into a single feature vector. This vector forms an efficient representation of the face area and can be used to measure similarities between images.

Automatic facial expression analysis is a fascinating and challenging problem and impacts important applications in several areas such as human and computer interaction and data-driven animation.

Deriving a facial representation from original face images is an essential step for successful facial expression recognition method. In this paper, we evaluate facial representation predicated on statistical local features, Local Binary Patterns, for facial expression recognition.

Various machine learning methods are systematically examined on several databases. Broad experiments illustrate that LBP features are effective and efficient for facial expression recognition.

The area binary pattern (LBP) was originally designed for texture description. It’s invariant to monotonic grey-scale transformations which are essential for texture description and analysis for the reason of computational simplicity processing of image in real-time is possible. With LBP it’s possible to explain the texture and model of an electronic digital image. This is completed by dividing a picture into several small regions from which the features are extracted.

These features contain binary patterns that describe the environmental surroundings of pixels in the regions. The features that are formed from the regions are concatenated into a single feature histogram, which describes to forms a representation of the image. Images will then be compared by measuring the similarity (distance) between their histograms.

According a number of studies face recognition utilising the LBP method provides positive results, both with regards to speed and discrimination performance.

Due to the way the texture and model of images is described, the technique is apparently quite robust against face images with different facial expressions, different lightening conditions, aging of persons and image rotation. Hence it is good for face recognition purposes.

8.6 Data Dictionary

Field Name	Data Type	Length	Constraint	Description
Roll_no	Int	3	Primary key	Student roll no
Name	Varchar	20	Not null	Name of student
Date	Date	10	Not null	Date of the attendance
Time	Time	10	Not null	Time of the attendance
Attendance	Varchar	7	Present or Absent	Attendance of a student
Images	.pgm	100	Size must be of 11KB	Images of students

8.7 Data Flow Diagrams

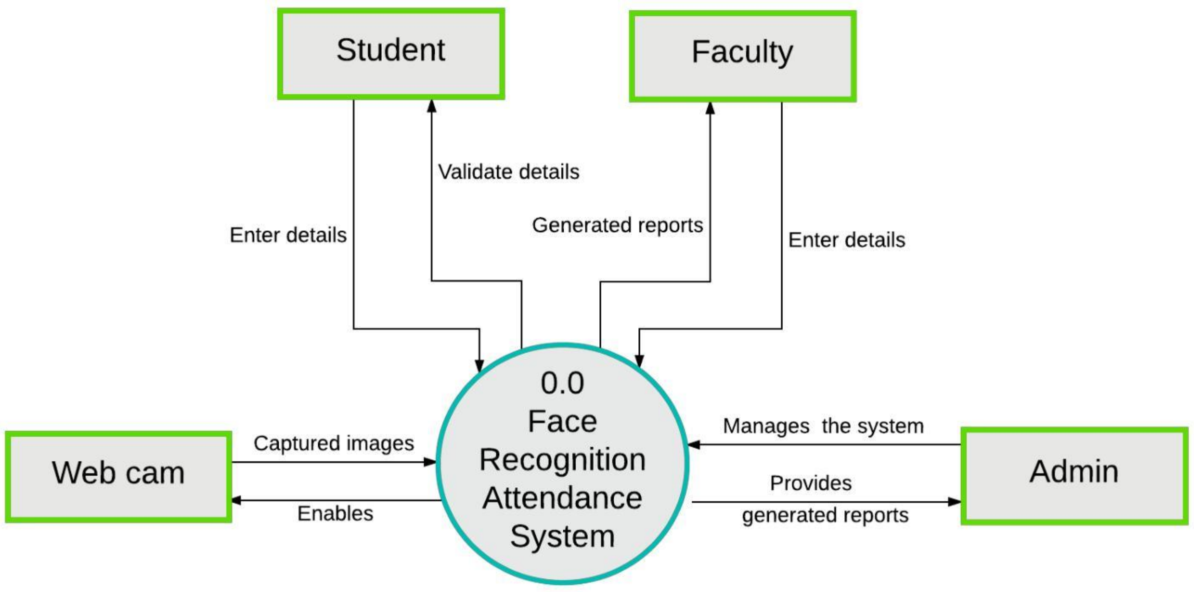


Figure 18 DFD: Context Level

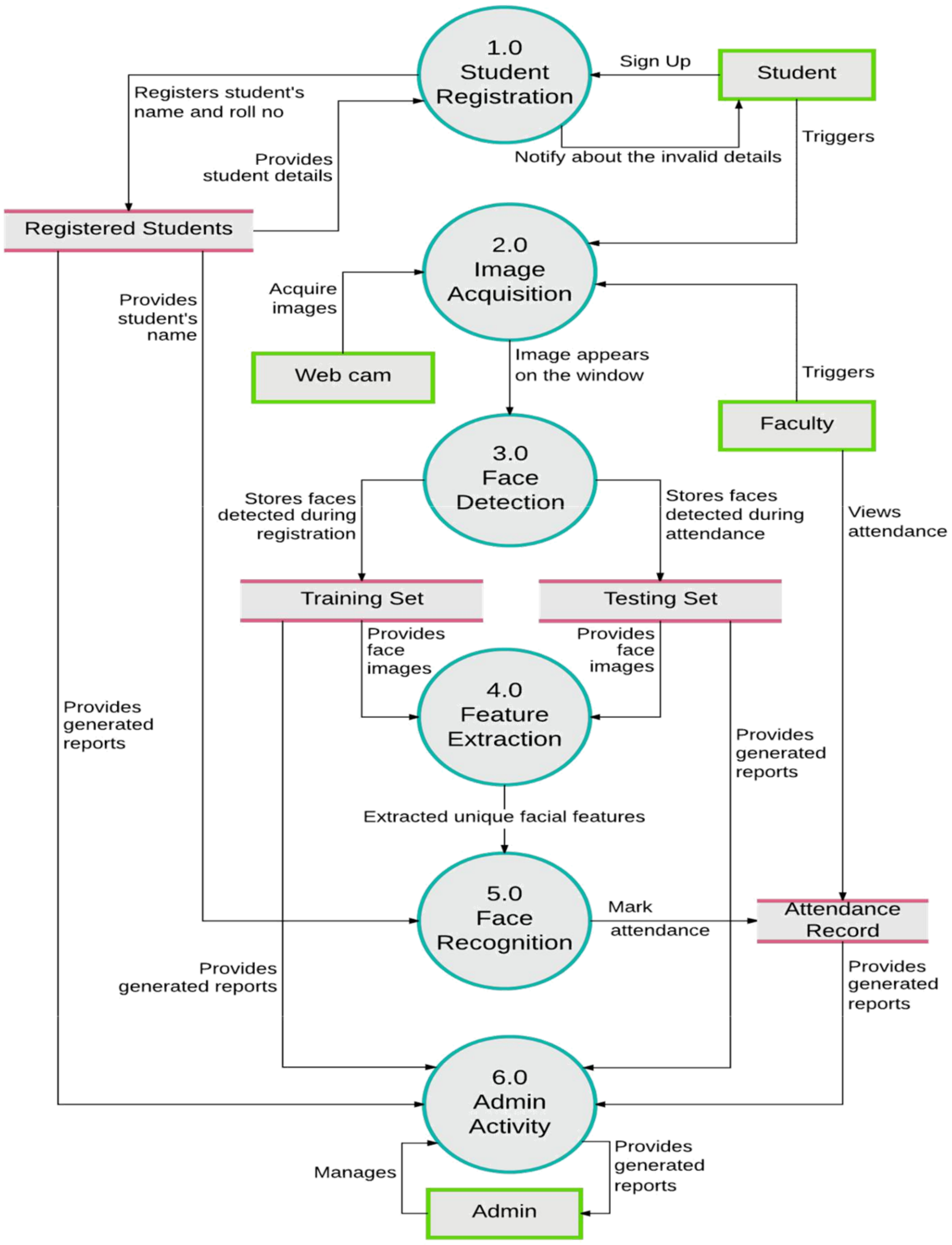


Figure 19 DFD: Level 0

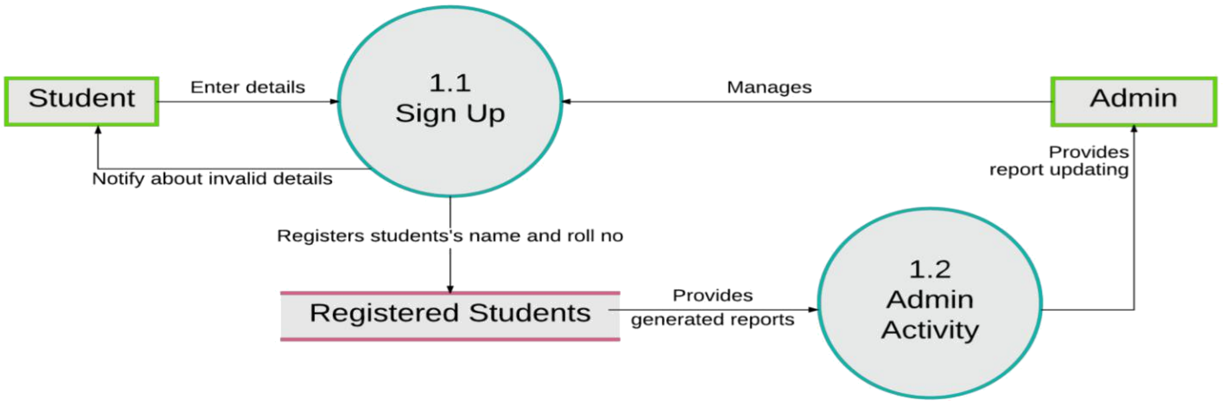


Figure 20 DFD: Level 1 (1.0 Student Registration)

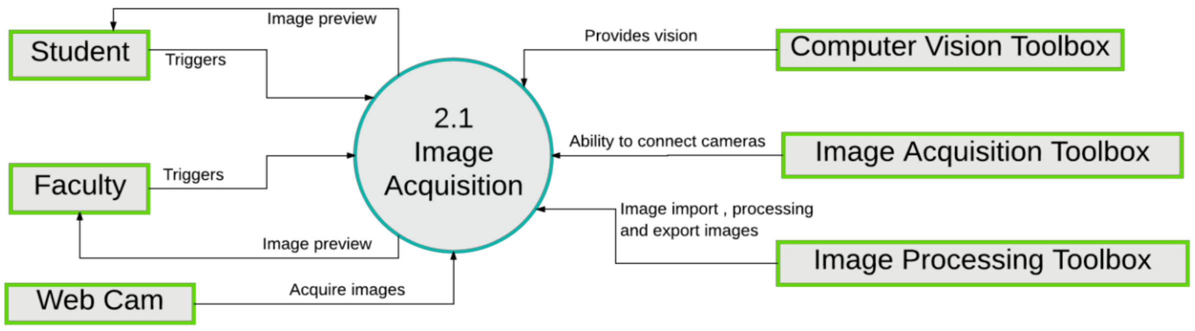


Figure 21 DFD: Level 1 (2.0 Image Acquisition)

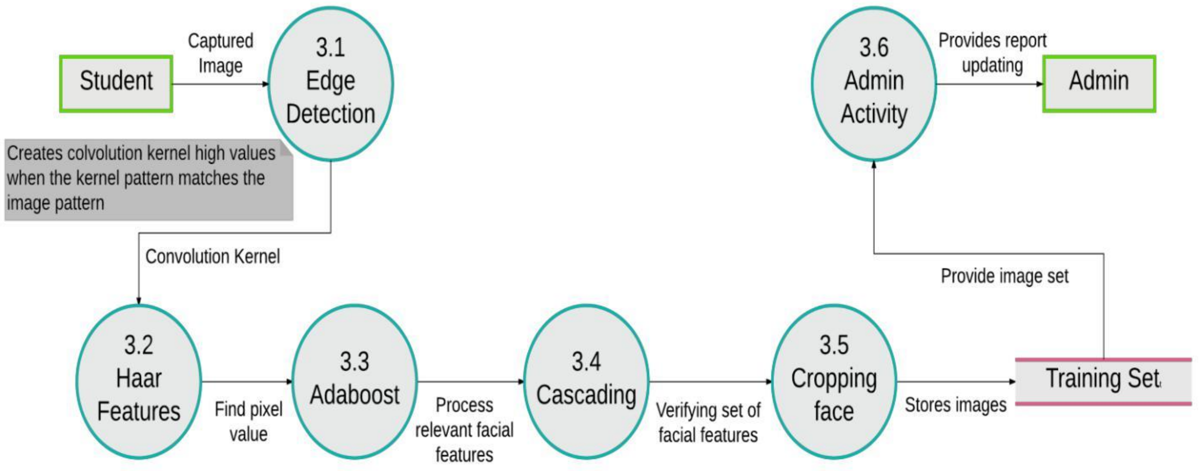


Figure 22 DFD: Level 1 (3.0 Face Detection)

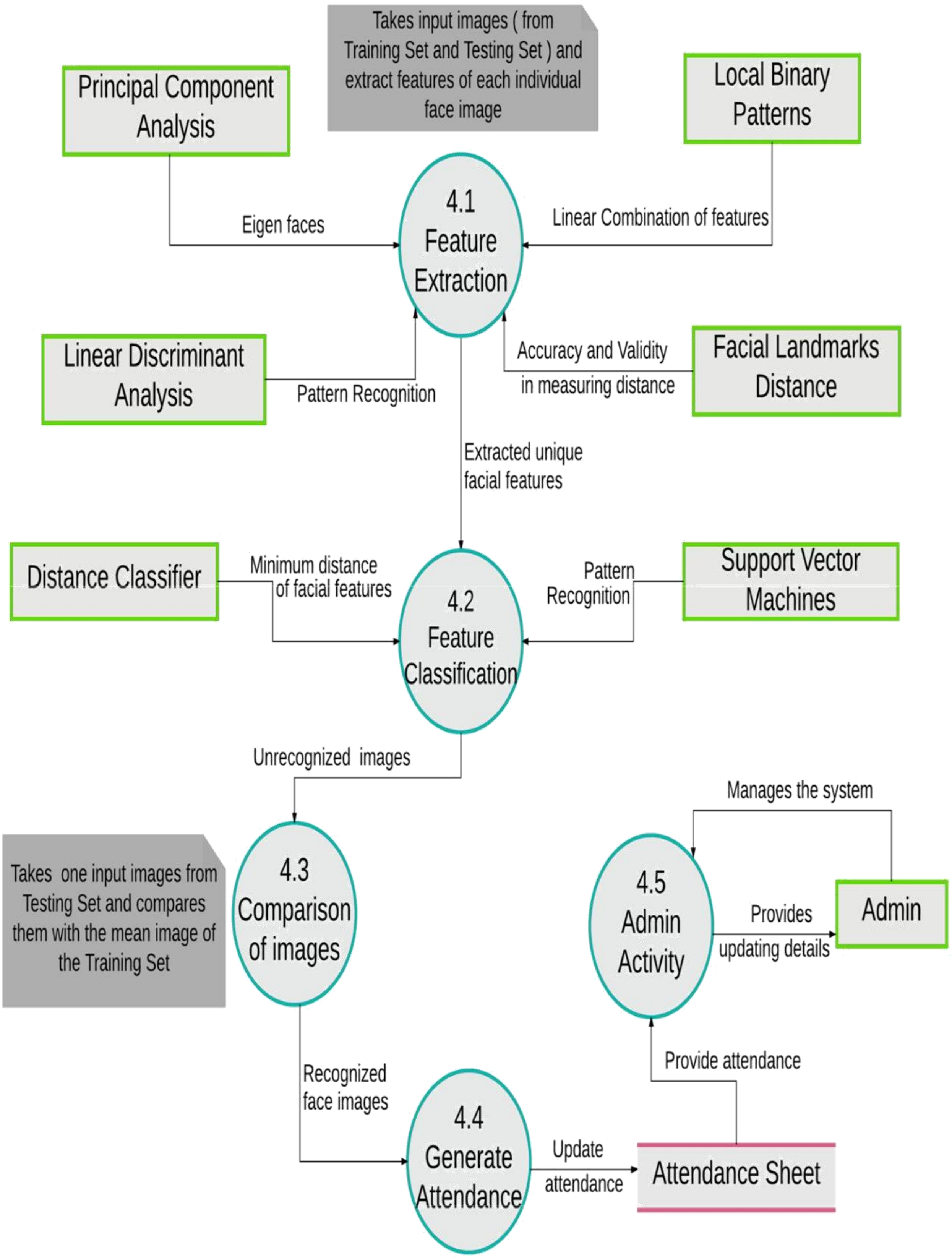

Takes input images (from Training Set and Testing Set) and extract features of each individual face image
Principal Component Analysis
Local Binary Patterns
Eigen faces
4.1 Feature Extraction
Linear Combination of features
Linear Discriminant Analysis
Pattern Recognition
Accuracy and Validity in measuring distance
Facial Landmarks Distance
Extracted unique facial features
Distance Classifier
Minimum distance of facial features
4.2 Feature Classification
Pattern Recognition
Support Vector Machines
Unrecognized images
Manages the system
Takes one input images from Testing Set and compares them with the mean image of the Training Set
4.3 Comparison of images
4.5 Admin Activity
Provides updating details
Admin
Provide attendance
Recognized face images
4.4 Generate Attendance
Update attendance
Attendance Sheet

Figure 23 DFD: Level 1 (4.0 Face Recognition)

CHAPTER 9

DATABASE DESIGN

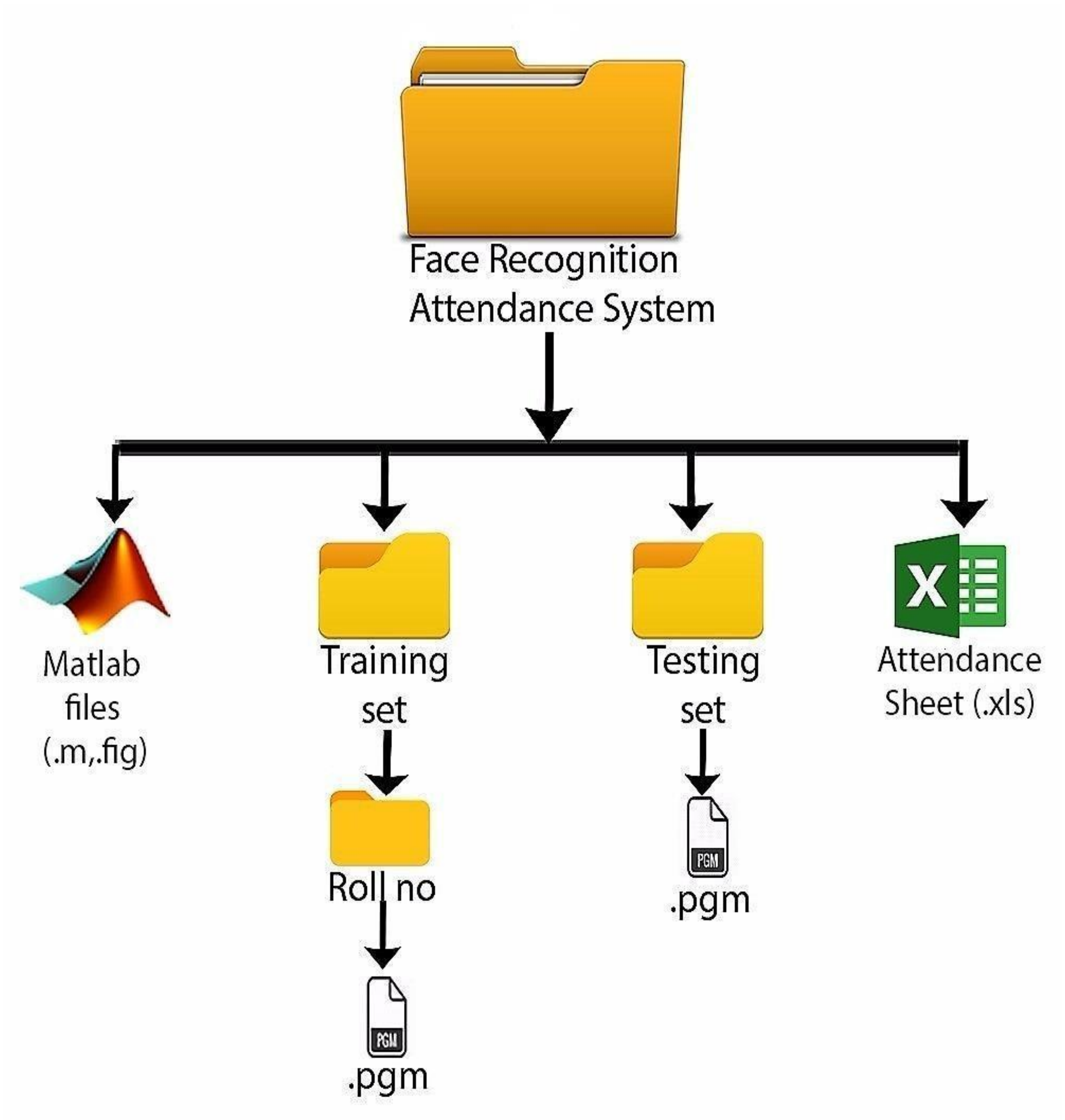


Figure 24 Database Design

CHAPTER 10
IMPLEMENTATION

System Flowchart

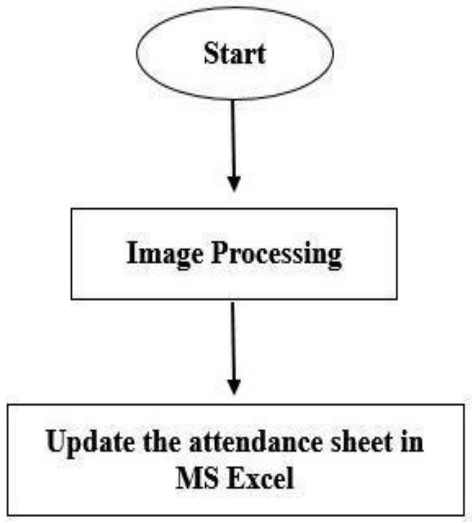
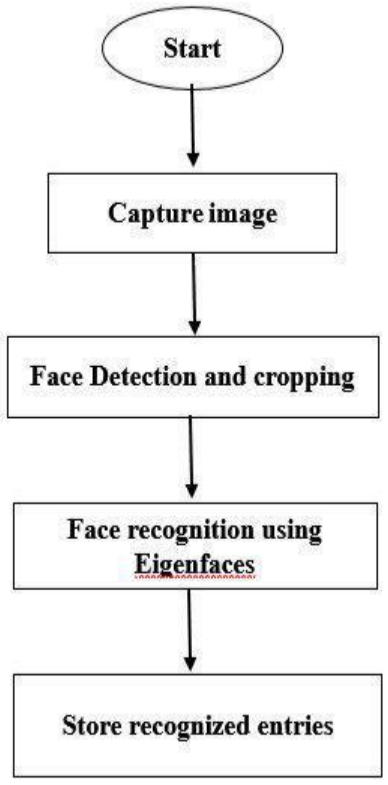


Image Processing



Start
Capture image
Face Detection and cropping
Face recognition using Eigenfaces
Store recognized entries

System Pre-Requisites

The first step in implementing the system is to create a database of enrolled students' database. In actual implementation this step must be a part of the admission process where we collect the necessary information of the students.

This set of images is referred to as train database for the algorithm. The facial recognition algorithm (here we use Eigenfaces method), then uses the database to calculate the eigenfaces for face recognition.

In our project we have created a function 'training.m' for this purpose. This function works as follows-

¬ Capture the image of the student

¬ Using the function 'visionCascadeObjectdetector' of the Computer vision toolbox, detect the face from the image. This function works on the basis of Viola-Jones algorithm.

¬ The detected faces are cropped and saved in the database.

This function must be in the folder where the main code is saved. Also, the train database is saved in the folder 'Train Database' which is also stored in the same folder for best results. After this step the system is ready for recording the attendance of the registered students.

Image Processing

This is the most important part of the system since it is based in the concept of image processing itself. This process is explained the sequence of its occurrence in the flowchart under the title image processing.

Capture Image

The image of the classroom is captured such that the faces of all the students are captured efficiently. This image is used for further processing of our algorithm. We have used the laptop camera with a resolution of 1366x768 itself since for the prototype this resolution is enough. For more accurate processing of a larger classroom, we need to use camera with higher resolution.

Face Detection and Cropping

The image captured image is read in MATLAB. The image is nothing but a matrix of numbers which correspond to the pixel values. The software doesn’t know where in this collection of numbers the faces are present, which are the input for our algorithm. Thus, face detection performs this task.

We use the function ‘vision. CascadeObjectDetector ()’ of the Computer Vision Toolbox for the same. This function detects the face based on the Viola-Jones algorithm whose description is given in the appendix. This sequence of steps in this algorithm is as follows.

3. Read the image captured in the previous step

- The faces are detected from the above image as explained earlier

4. We crop the area of the image where the faces are marked and saved into a folder as individual image files in JPEG format.

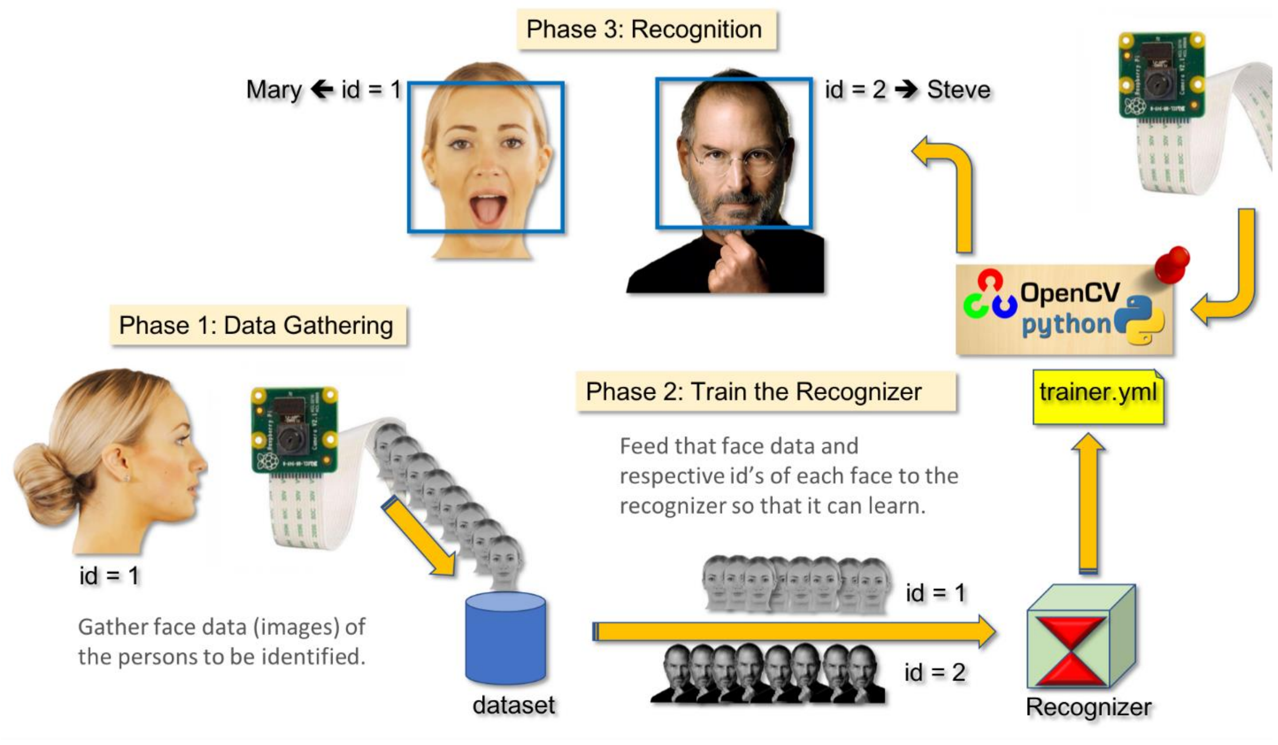
The algorithm detects all the faces clearly visible in the captured image of the classroom. Each student needs to be in an upright right position to avoid exclusion of their presence by the system.

To create a complete project on Face Recognition, we must work on 3 very distinct phases:

1. Face Detection and Data Gathering
2. Train the Recognizer

3. Face Recognition

The below block diagram resumes those phases:


Phase 3: Recognition
Mary ← id = 1
id = 2 → Steve
OpenCV
python
Phase 1: Data Gathering
Phase 2: Train the Recognizer
trainer.yml
Feed that face data and respective id's of each face to the recognizer so that it can learn.
id = 1
Gather face data (images) of the persons to be identified.
id = 1
id = 2
dataset
Recognizer

Step: 1 First we have to install OpenCV module which is a **cross-platform library using which we can develop real-time computer vision applications**. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

- In this project am using and working on jupyter notebook and for this platform first we have to install anaconda.
- Download and **install anaconda** environment Python 3.7: Download: https://www.**anaconda**.com/download/#windows.
- Open **Anaconda** Prompt. Start Menu / Anaconda3 / **Anaconda** Prompt.
- In **Anaconda** Prompt, type **commands to install** necessary libraries: pip **install OpenCV**-python==3.4.2.17. ...

- Run your python program.

Step: 2 **Testing your camera**

Once you have OpenCV installed let's test to confirm that your camera is working properly.

Enter the below Python code on your IDE:

```
import NumPy as np

import cv2


cap = cv2.VideoCapture(0)

cap.set (3,640) # set Width

cap.set (4,480) # set Height


while (True):

    ret, frame = cap. read()

    frame = cv2.flip(frame, -1) # Flip camera vertically

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)


    cv2.imshow('frame', frame)

    cv2.imshow('gray', gray)


    k = cv2.waitKey(30) & 0xff

    if k == 27: # press 'ESC' to quit
```

```
        break
cap.release()
cv2.destroyAllWindows()
```

The above code will capture the video stream that will be generated by your webcam, displaying both, in BGR color and Gray mode.

Step: 3 **Face Detection**

The most basic task on Face Recognition is of course, "Face Detecting". Before anything, you must "capture" a face (Phase 1) in order to recognize it, when compared with a new face captured on future (Phase 3).

The most common way to detect a face (or any objects), is using the "Haar Cascade classifier"

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. The good news is that OpenCV comes with a trainer as well as a detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one.

If you do not want to create your own classifier, OpenCV already contains many pre-trained classifiers for face, eyes, smile, etc. Those XML files can be download from haarcascades directory.

```
import NumPy as np
import cv2
faceCascade = cv2.CascadeClassifier('Cascades/haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0)
cap.set (3,640) # set Width
cap.set (4,480) # set Height
while True:
    ret, img = cap.read()
    img = cv2.flip(img, -1)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.2,
        minNeighbors=5,
        minSize= (20, 20)
    )
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (255,0,0),2)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = img[y:y+h, x:x+w]
    cv2.imshow('video',img)
    k = cv2.waitKey(30) & 0xff
    if k == 27: # press 'ESC' to quit
        break
cap.release()
cv2.destroyAllWindows()
```

Now we must call our classifier function, passing it some very important parameters, as scale factor, number of neighbors and minimum size of the detected face.

```
faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.2,
        minNeighbors=5,
        minSize=(20, 20)
    )
```

Where,

- **gray** is the input grayscale image.
- **scale Factor** is the parameter specifying how much the image size is reduced at each image scale. It is used to create the scale pyramid.
- **minNeighbors** is a parameter specifying how many neighbors each candidate rectangle should have, to retain it. A higher number gives lower false positives.
- **minSize** is the minimum rectangle size to be considered a face.

The function will detect faces on the image. Next, we must "mark" the faces in the image, using, for example, a blue rectangle. This is done with this portion of the code:

```
for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
```

If faces are found, it returns the positions of detected faces as a rectangle with the left up corner (x,y) and having "w" as its Width and "h" as its Height ==> (x,y,w,h). Please see the above picture.

Once we get these locations, we can create an "ROI" (drawn rectangle) for the face and present the result with *imshow()* function.

You can also include classifiers for "eyes detection" or even "smile detection". On those cases, you will include the classifier function and rectangle draw inside the face loop, because would be no sense to detect an eye or a smile outside of a face.

Step: 4 **Data Gathering**

let's start the first phase of our project. What we will do here, is starting from last step (Face Detecting), we will simply create a dataset, where we will store for each id, a group of photos in gray with the portion that was used for face detecting.

First, create a directory where you develop your project, for example, FacialRecognitionProject:

```
mkdir FacialRecognitionProject
```

In this directory, besides the 3 python scripts that we will create for our project, we must have saved on it the Facial Classifier.

Next, create a subdirectory where we will store our facial samples and name it "dataset":

```
mkdir dataset
```

Enter the below python code for 01_face_dataset.py on your IDE.

```python
import cv2

import os

cam = cv2.VideoCapture(0)

cam.set(3, 640) # set video width

cam.set(4, 480) # set video height

face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# For each person, enter one numeric face id

face_id = input('\n enter user id end press <return> ==>  ')

print("\n [INFO] Initializing face capture. Look the camera and wait ...")

# Initialize individual sampling face count

count = 0


while(True):

    ret, img = cam.read()

    img = cv2.flip(img, -1) # flip video image vertically

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    faces = face_detector.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:

        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
        count += 1

        # Save the captured image into the datasets folder

        cv2.imwrite("dataset/User." + str(face_id) + '.' + str(count) + ".jpg",
gray[y:y+h,x:x+w])

        cv2.imshow('image', img)

    k = cv2.waitKey(100) & 0xff # Press 'ESC' for exiting video

    if k == 27:

        break

    elif count >= 30: # Take 30 face sample and stop video

         break

# Do a bit of cleanup

print("\n [INFO] Exiting Program and cleanup stuff")

cam.release()

cv2.destroyAllWindows()
```

The code is very similar to the code that we saw for face detection. What we added, was an "input command" to capture a user id, that should be an integer number (1, 2, 3, etc)

```
face_id = input('\n enter user id end press  ==>  ')
```

And for each one of the captured frames, we should save it as a file on a "dataset" directory:

```
cv2.imwrite("dataset/User." + str(face_id) + '.' + str(count) + ".jpg", gray[y:y+h,x:x+w])
```

Note that for saving the above file, you must have imported the library "os". Each file's name will follow the structure:

```
User.face_id.count.jpg
```

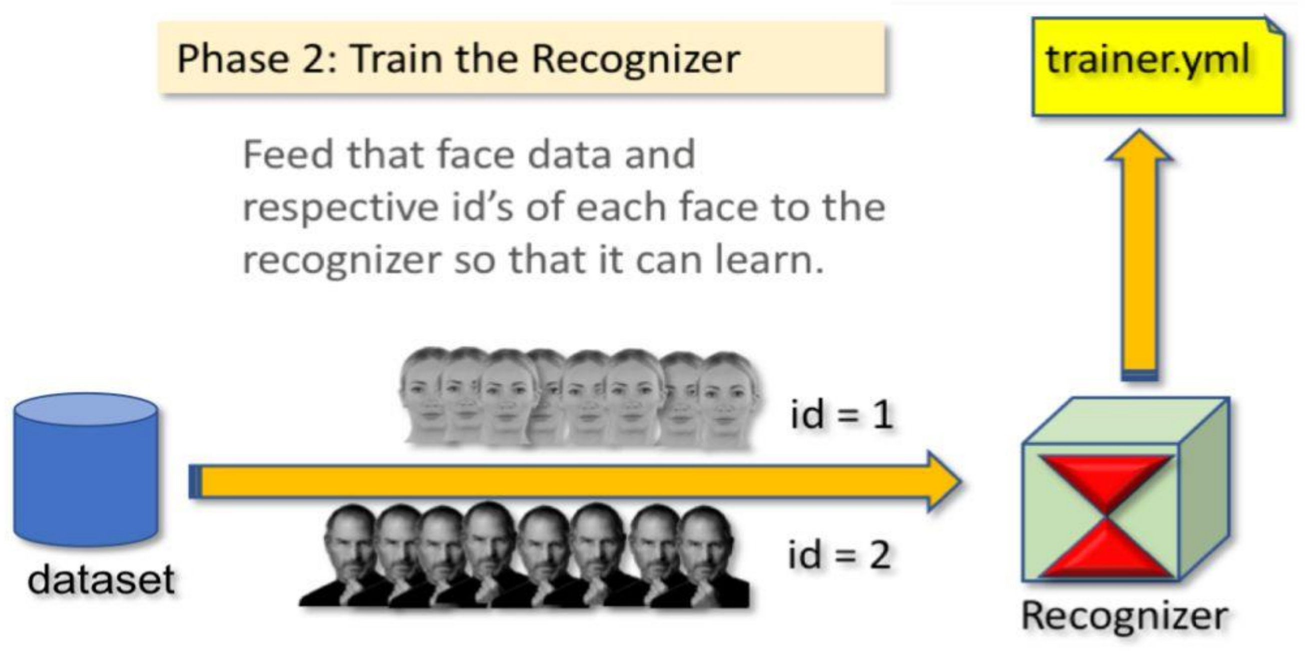
For example, for a user with a face_id = 1, the 4th sample file on dataset/ directory will be something like:

```
User.1.4.jpg
```

On my code, I am capturing 30 samples from each id. You can change it on the last "elif". The number of samples is used to break the loop where the face samples are captured.

Run the Python script and capture a few Ids. You must run the script each time that you want to aggregate a new user (or to change the photos for one that already exists).

Step: 5 **Trainer**


Phase 2: Train the Recognizer
trainer.yml
Feed that face data and respective id's of each face to the recognizer so that it can learn.
id = 1
id = 2
dataset
Recognizer

On this second phase, we must take all user data from our dataset and "trainer" the OpenCV Recognizer. This is done directly by a specific OpenCV function. The result will be a .yml file that will be saved on a "trainer/" directory.

So, let's start creating a subdirectory where we will store the trained data:

```
mkdir trainer
```

Enter the below python code for 02_face_training.py on your IDE

```
import cv2

import numpy as np

from PIL import Image

import os

# Path for face image database

path = 'dataset'


recognizer = cv2.face.LBPHFaceRecognizer_create()

detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");

# function to get the images and label data

def getImagesAndLabels(path):

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]

    faceSamples=[]

    ids = []

    for imagePath in imagePaths:
        PIL_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')
        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)
        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)
    return faceSamples,ids
print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))


# Save the model into trainer/trainer.yml
recognizer.write('trainer/trainer.yml') # recognizer.save() worked on Mac, but not on Pi
# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained. Exiting Program".format(len(np.unique(ids))))
```

Confirm if you have the PIL library installed. If not, run the below command in Terminal:

```
pip install pillow
```

We will use as a recognizer, the LBPH (LOCAL BINARY PATTERNS HISTOGRAMS) Face Recognizer, included on OpenCV package. We do this in the following line:

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
```

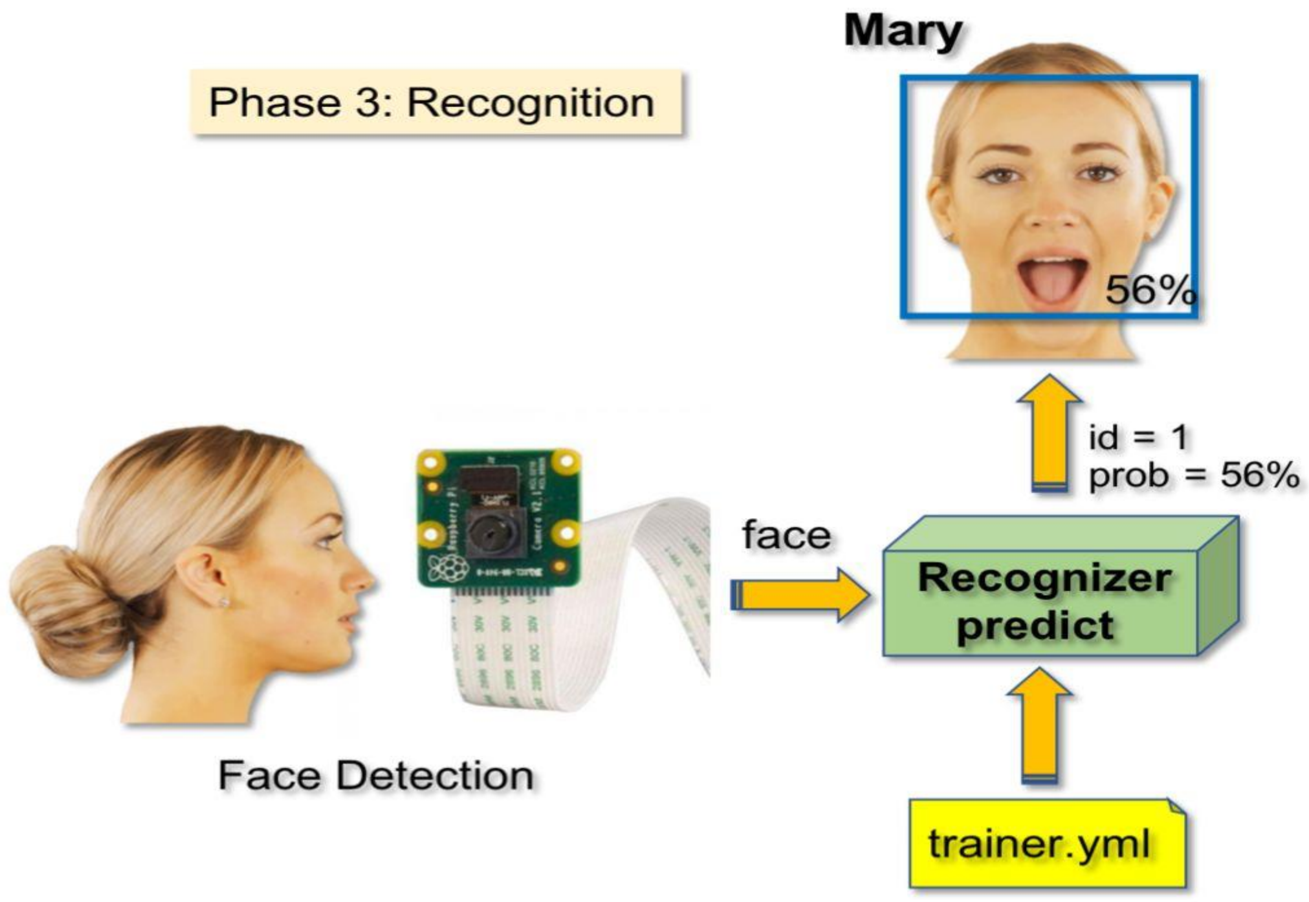
The function "getImagesAndLabels (path)", will take all photos on directory: "dataset/", returning 2 arrays: "Ids" and "faces". With those arrays as input, we will "train our recognizer":

```
recognizer.train(faces, ids)
```

As a result, a file named "trainer.yml" will be saved in the trainer directory that was previously created by us.

Every time that you perform Phase 1, Phase 2 must also be run.

Step: 6 **Recognizer**



Now, we reached the final phase of our project. Here, we will capture a fresh face on our camera and if this person had his face captured and trained before, our recognizer will make a "prediction" returning its id and an index, shown how confident the recognizer is with this match.

Enter the below python code for 03_face_Recognition.py on your IDE

```
import cv2

import numpy as np

import os


recognizer = cv2.face.LBPHFaceRecognizer_create()

recognizer.read('trainer/trainer.yml')

cascadePath = "haarcascade_frontalface_default.xml"

faceCascade = cv2.CascadeClassifier(cascadePath);


font = cv2.FONT_HERSHEY_SIMPLEX


#iniciate id counter

id = 0


# names related to ids: example ==> Marcelo: id=1,  etc

names = ['None', 'Marcelo', 'Paula', 'Ilza', 'Z', 'W']
```

```
# Initialize and start realtime video capture

cam = cv2.VideoCapture(0)

cam.set(3, 640) # set video widht

cam.set(4, 480) # set video height


# Define min window size to be recognized as a face

minW = 0.1*cam.get(3)

minH = 0.1*cam.get(4)


while True:

    ret, img =cam.read()

    img = cv2.flip(img, -1) # Flip vertically

    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)


    faces = faceCascade.detectMultiScale(

        gray,

        scaleFactor = 1.2,

        minNeighbors = 5,

        minSize = (int(minW), int(minH)),

       )


    for(x,y,w,h) in faces:
        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

        id, confidence = recognizer.predict(gray[y:y+h,x:x+w])


        # Check if confidence is less them 100 ==> "0" is perfect match

        if (confidence < 100):

            id = names[id]

            confidence = "  {0}%".format(round(100 - confidence))

        else:

            id = "unknown"

            confidence = "  {0}%".format(round(100 - confidence))


        cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255), 2)

        cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)


    cv2.imshow('camera',img)


    k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video

    if k == 27:

        break


# Do a bit of cleanup

print("\n [INFO] Exiting Program and cleanup stuff")
```

```
cam.release()

cv2.destroyAllWindows()
```

We are including here a new array, so we will display "names", instead of numbered ids:

```
names = ['None', 'Monika', 'Kartik', 'Ayesha', 'Z', 'W']
```

So, for example: Monika will the user with id = 1; Kartik: id=2, etc.

Next, we will detect a face, same we did before with the haasCascade classifier. Having a detected face we can call the most important function in the above code:

```
id, confidence = recognizer.predict(gray portion of the face)
```

The recognizer.predict (), will take as a parameter a captured portion of the face to be analyzed and will return its probable owner, indicating its id and how much confidence the recognizer is in relation with this match.

Note that the confidence index will return "zero" if it will be cosidered a perfect match

And at last, if the recognizer could predict a face, we put a text over the image with the probable id and how much is the "probability" in % that the match is correct ("probability" = 100 - confidence index). If not, an "unknow" label is put on the face.

CHAPTER 11

CONCLUSION

The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made by the researcher were reliable. The system with manual face detection and automatic face recognition did not have a recognition accuracy over 90%, due to the limited number of eigenfaces that were used for the PCA transform. This system was tested under very robust conditions in this experimental study and it is envisaged that real-world performance will be far more accurate. The fully automated frontal view face detection system displayed virtually perfect accuracy and in the researcher's opinion further work need not be conducted in this area.

The fully automated face detection and recognition system was not robust enough to achieve a high recognition accuracy. The only reason for this was the face recognition subsystem did not display even a slight degree of invariance to scale, rotation or shift errors of the segmented face image. This was one of the system requirements identified in section 2.3. However, if some sort of further processing, such as an eye detection technique, was implemented to further normalise the segmented face image, performance will increase to levels comparable to the manual face detection and recognition system. Implementing an eye detection technique would be a minor extension to the implemented system and would not require a great deal of additional research.All other implemented systems displayed commendable results and reflect well on the deformable template and Principal Component Analysis strategies.

In this system we have implemented an attendance system for a lecture, section or laboratory by which lecturer or teaching assistant can record students’ attendance. It saves time and effort, especially if it is a lecture with huge number of students. Automated Attendance System has been envisioned for the purpose of reducing the drawbacks in the traditional (manual) system. This attendance system demonstrates the use of image processing techniques in classroom. This system can not only merely help in the attendance system, but also improve the goodwill of an institution. The Automated Classroom Attendance System helps in increasing the accuracy and speed ultimately achieve the high-precision real-time attendance to meet the need for automatic classroom evaluation.

At the end, the system not only resolve troubles that exist in the old model but also provide convenience to the user to access the information collected which perfected the existence of technology to assist human’s needs.

CHAPTER 12

APPENDICES

FACE DETECTION:

```
clear all

clr

%Detect objects using Viola-Jones Algorithm



%To detect Face

FDetect = vision.CascadeObjectDetector;



%Read the input image

I = imread('HarryPotter.jpg');


%Returns Bounding Box values based on number of objects
    BB = step(FDetect,I);

figure,

imshow(I); hold on

for i = 1:size(BB,1)

rectangle('Position',BB(i,:),'LineWidth',5,'LineStyle','-
    ','EdgeColor','r');

end
```

```
title('Face Detection');

hold off;
```

The step(Detector,I) returns Bounding Box value that contains [x,y,Height,Width] of the objects of interest.

```
BB =

     52     38     73    73
    379     84     71    71
    198     57     72    72
```

NOSE DETECTION:

```
%To detect Nose

NoseDetect = vision.CascadeObjectDetector('Nose','MergeThreshold',16);

BB=step(NoseDetect,I);

figure,

imshow(I); hold on

for i = 1:size(BB,1)

rectangle('Position',BB(i,:),'LineWidth',4,'LineStyle','-
    ','EdgeColor','b');

end
```

```
title('Nose Detection');

hold off;
```

EXPLANATION:

To denote the object of interest as 'nose', the argument 'Nose' is passed.

```
vision.CascadeObjectDetector('Nose','MergeThreshold',16);
```

The default syntax for Nose detection :

```
vision.CascadeObjectDetector('Nose');
```

Based on the input image, we can modify the default values of the parameters passed to **vision.CascaseObjectDetector.** Here the default value for 'MergeThreshold' is 4.

When default value for 'MergeThreshold' is used, the result is not correct.

Here there are more than one detection on Hermione.

To avoid multiple detection around an object, the 'MergeThreshold' value can be overridden.

MOUTH DETECTION:

```
%To detect Mouth

MouthDetect = vision.CascadeObjectDetector('Mouth','MergeThreshold',16);

BB=step(MouthDetect,I);
```

```
figure,

imshow(I); hold on

for i = 1:size(BB,1)

  rectangle('Position',BB(i,:),'LineWidth',4,'LineStyle','-
','EdgeColor','r');

end

title('Mouth Detection');

hold off;
```

EYE DETECTION:

```
%To detect Eyes

EyeDetect = vision.CascadeObjectDetector('EyePairBig');

%Read the input Image

        I  = imread('harry_potter.jpg');

        BB=step(EyeDetect,I);

figure,imshow(I);

rectangle('Position',BB,'LineWidth',4,'LineStyle','-','EdgeColor','b');

title('Eyes Detection');
```

```
Eyes=imcrop(I,BB);

figure,imshow(Eyes);
```

FACE RECOGNITION

```
function closeness=recognition(input_image,U,R);



            %By L.S. Balasuriya

      %     Returns closeness of known face vectors in
      R to unknown image input_image % R contains
      the Eigenfaces


%********image to vector
**********************************************


vinput=reshape(input,[10000 1]);


%********recognition
*************************************************


facespace=voutput'*U;


%********Eucleadian distance
*****************************************
```

```
[p,ignor]=size(R);

distance_vecs=R-repmat(facespace,[p 1]);

distance=sum(abs(distance_vecs)')';

%********order of closeness to unknown face ****************************

[ignor,closeness]=sort(distance);
```

CHAPTER 13

REFERENCES

[1] M. A. Turk and A. P. Pentland, "Face Recognition Using Eigenfaces," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 586–591. 1991.

[2] A. J. Goldstein, L. D. Harmon, and A. B. Lesk, "Identification of Human Faces," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, vol. 59, pp 748 – 760, May 1971.

[3] T. Ahonen, A. Hadid, M. Peitikainen, Face recognition with local binary patterns. "In Proc. of European Conference of Computer Vision", 2004.

[4] M. H. Yang, N. Ahuja, and D. Kriegmao, "Face recognition using kernel eigenfaces," IEEE International Conference on Image Processing, vol. 1, pp. 10-13, Sept. 2000.

[5] Y. Zhang and C. Liu, "Face recognition using kemel principal component analysis and genetic algorithms," IEEE Workshop on Neural Networks for Signal Processing, pp. 4-6 Sept. 2002.

[6] M. A. Fischler and R. A. Elschlager, "The Representation and Matching of Pictorial Structures," IEEE Transaction on Computer, vol. C22, pp. 67-92, 1973.

[7] P. Sinha, B. Balas, Y. Ostrovsky, and R. Russell, "Face Recognition by Humans: Nineteen Results All Computer Vision Researchers Should Know About," in Proceedings of the IEEE, vol. 94, Issue 11, 2006.

[8] A. T. Acharya and A. Ray, Image Processing: Principles and Applications, New York: Wiley, 2005.

[9] T. D. Russ, M. W. Koch, and C. Q. Little, "3D Facial Recognition: A Quantitative Analysis," 38th Annual 2004 International Carnahan Conference on Security Technology, 2004.

[10] S. S. R. Abibi, “Simulating evolution: connectionist metaphors for studying human cognitive behaviour,” in Proceedings TENCON 2000, vol. 1 pp 167-173, 2000.

[11] B. Tacacs and H. Wechsler. Face recognition using binary image metrics. In Proceedings, Third International Conference on Automatic Face and Gesture Recognition, pages 294–299, April 1998.

[12] Y. Cui, J. S. Jin, S. Luo, M. Park, and S. S. L. Au, “Automated Pattern Recognition and Defect Inspection System,” in proc. 5 th International Conference on Computer Vision and Graphical Image, vol. 59, pp. 768 – 773, May 1992.