Realtime Facial Detection and Identification System

Project report submitted
in partial fulfillment of the requirement for the degree of

Bachelor of Technology

By

Prashant Mishra (1400000225)

Under the Guidance of
Prof. Sunita S. Dhotre

DEPARTMENT OF COMPUTER ENGINEERING

BHARATI VIDYAPEETH DEEMED TO BE UNIVERSITY,
COLLEGE OF ENGINEERING, PUNE- 43
(2017-2018)

BHARATI VIDYAPEETH DEEMED TO BE UNIVERSITY,
COLLEGE OF ENGINEERING, PUNE- 43

CERTIFICATE

This is to certify that the project report titled **Realtime Facial Detection and Identification System**, has been carried out by the following student

1. Prashant Mishra

under the supervision of **Prof. Sunita S. Dhotre** in partial fulfillment of the degree of **BACHELOR OF TECHNOLOGY** in Computer Engineering of Bharati Vidyapeeth Deemed University, College of Engineering, Pune during the academic year 2017-2018.

Guide Project Coordinator Head of Department

Place:
Date:

BHARATI VIDYAPEETH DEEMED TO BE UNIVERSITY,

COLLEGE OF ENGINEERING, PUNE- 43

APPROVAL CERTIFICATE

This project report entitled **(Realtime Facial Detection and Identification System)** by **(Prashant Mishra)** is approved for the degree of **BACHELOR OF TECHNOLOGY**

Examiner Name & Sign

Guide's Name & Sign

Project Coordinator's Name & Sign

Head of Department

Place:
Date:

ACKNOWLEDGEMENTS

We wish to express our sincere thanks and gratitude to Prof. Sunita S. Dhotre, Professor, Department of Computer Science, Bharati Vidyapeeth Pune, for the stimulating discussions, in analysing problems associated with our project work and for guiding us throughout the project. Project meetings were highly informative. We express our warm and sincere thanks for the encouragement, untiring guidance and the confidence he had shown in us. We are immensely indebted for his invaluable guidance throughout our project. We also thank Prof. D.M Thakore, Head of Department, Department of Computer Science, Bharati Vidyapeeth Pune, for helping us out during the execution of our project. Finally, we would also like to thank our parents and friends for constant encouragement and help throughout the course of the project.

ABSTRACT

Face recognition is an important application of Image processing owing to its use in many fields like crowd identification, video conference, security measure, image analysis etc. Like all biometrics solutions, face recognition technology measures and matches the unique characteristics for the purposes of identification or authentication. Often leveraging a digital or connected camera, different algorithms and techniques can be used to detect faces in images and video, quantify their features, and then match them against stored templates in a database. Many techniques are being proposed, ranging from simple edge based algorithm to composite high level approaches utilizing advanced pattern recognition methods.

The algorithms presented in this report are Viola-Jones algorithm (Haar Cascade Classifier) and Principal Component Analysis (PCA) and are discussed in terms of technical approach and performance. The objective of this project is to develop an effective and real-time face recognition system based on OpenCV-Viola-Jones algorithm (Haar Cascade Classifier), Principal Component Analysis and Java servlet. This technology can used by Law enforcement agencies for monitoring, finding suspects and criminals on road traffic, railway stations, scan faces in CCTV footage, Border control deployments and can also be used as consumer application in schools, colleges, and corporate offices for attendances and security.

Keywords: *(Real time, Face Detection, Face Tracking, Face Recognition, OpenCV, PCA)*

TABLE OF CONTENTS

CHAPTER-1
INTRODUCTION

CHAPTER-1
INTRODUCTION

1.1 Introduction

Face recognition has been a sought-after problem of biometrics and it has a variety of applications in modern life. The problems of face recognition attract researchers working in biometrics, pattern recognition field and computer vision. An efficient face recognition system can be of great help in forensic sciences, identification for law enforcement, surveillance, authentication for banking and security system, and giving preferential access to authorized users i.e. access control for secured areas etc. The problem of face recognition has gained even more importance after the recent increase in the terrorism related incidents. Use of face recognition for authentication also reduces the need of remembering passwords and can provide a much greater security if face recognition is used in combination with other security measures for access control. The cost of the license for an efficient commercial Face recognition system is very high which shows the significant value of the problem. An automatic security system using facial biometrics would provide the needed solution This project aims to build an opensource facial recognition system which has wide implementation in security and surveillance and minimizing man power by using computation for security as much as possible. This application can be installed on the monitoring desktop taking input from cameras like (webcams, CCTVs, security camps, IP cameras,) detecting face real time.
Identifying it by matching images from the database and the same time save record with details like person name, place where he/she was seen and at what time, etc. Here, the application is using java run time environment, java servlet, jfoenix OpenCV library, haarcascades, lbpcascades Using this software, system perform multiple face detection, recognition, tracking suspect location and time by updating the log.

1.2 Problem Definition

Cameras enable users to record footage for later viewing, and to help nab criminals, and receive justice from the law. They cannot, however, stop a crime when it is in progress. They do not alert neighbors or the police like an alarm system would. This means that you will incur losses even as you run to the court, make insurance claims and reorder stolen inventory, which may no longer make you feel absolutely safe and even cause you to lose faith in them. People believe that a high number of cameras results in less crime. However, we know that CCTV cameras can record the mishappenings but can’t prevent or alert an undesirable event.

1.3 Objective, Goal and scope of the research work

A security system using facial biometrics implemented with the cameras or CCTV in public places, banks, or in homes would provide the needed solution This project aims to build an opensource facial recognition system which has wide implementation in

security and surveillance and minimizing man power by using computation for security as much as possible. This application can be installed on the monitoring desktop taking input from cameras like (webcams, CCTVs, security camps, IP cameras,) detecting and recognizing faces in real time.
Identifying it by matching images from the database and the same time save record with details like person name, place where he/she was seen and at what time, etc. Here, the application is using java run time environment, java servlet, jfoenix OpenCV library, haarcascades, lbpcascades Using this software, system perform multiple face detection, recognition, tracking suspect location and time by updating the log.

Further implementation can be to send notification to owner, authority, neighbors or triggering alarm for intrusion, if it’s any unknown, criminal or any convicted person and that could lead to stopping crime before it actually happens.

This system has unlimited scopes as it can be implement in public places like Railway Stations, Airports, Sea Ports, and Market Places stopping criminal and terrorist suspects from causing undesirable event.
It can be implemented in homes, shops, or any personal properties preventing trespassers or troublesome people from causing any harm, damage or burglary by triggering alarm or notification.

CHAPTER-2
BACKGROUND

CHAPTER-2

BACKGROUND

2.1 Facial Detection

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene. Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class. Examples include upper torsos, pedestrians, and cars.

Face-detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit. Image matches with the image stores in database. Any facial feature changes in the database will invalidate the matching process.

A reliable face-detection approach based on the genetic algorithm and the eigen-face technique.

Firstly, the possible human eye regions are detected by testing all the valley regions in the gray-level image. Then the genetic algorithm is used to generate all the possible face regions which include the eyebrows, the iris, the nostril and the mouth corners.

Each possible face candidate is normalized to reduce both the lightning effect, which is caused by uneven illumination; and the shirring effect, which is due to head movement. The fitness value of each candidate is measured based on its projection on the eigen-faces. After a number of iterations, all the face candidates with a high fitness value are selected for further verification. At this stage, the face symmetry is measured and the existence of the different facial features is verified for each face candidate.

2.2 Facial Recognition

A face recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a face database.
It is typically used in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Recently, it has also become popular as a commercial identification and marketing tool.
Some face recognition algorithms identify facial features by extracting landmarks, or features, from an image of the subject's face. For example, an algorithm may analyze the relative position, size, and/or shape of the eyes, nose, cheekbones, and jaw. These features are then used to search for other images with matching features. Other algorithms normalize a gallery of face images and then compress the face data, only saving the data in the image that is useful for face recognition. A probe image is then

compared with the face data. One of the earliest successful systems is based on template matching techniques applied to a set of salient facial features, providing a sort of compressed face representation.
Recognition algorithms can be divided into two main approaches, geometric, which looks at distinguishing features, or photometric, which is a statistical approach that distills an image into values and compares the values with templates to eliminate variances.
Popular recognition algorithms include principal component analysis using eigenfaces, linear discriminant analysis, elastic bunch graph matching using the Fisher face algorithm, the hidden Markov model, the multilinear subspace learning using tensor representation, and the neuronal motivated dynamic link matching.

Face recognition is an easy task for humans. Experiments in [Tu06] have shown, that even one to three days old babies are able to distinguish between known faces. So how hard could it be for a computer? It turns out we know little about human recognition to date. Are inner features (eyes, nose, mouth) or outer features (head shape, hairline) used for a successful face recognition? How do we analyze an image and how does the brain encode it? It was shown by David Hubel and Torsten Wiesel, that our brain has specialized nerve cells responding to specific local features of a scene, such as lines, edges, angles or movement. Since we don't see the world as scattered pieces, our visual cortex must somehow combine the different sources of information into useful patterns. Automatic face recognition is all about extracting those meaningful features from an image, putting them into a useful representation and performing some kind of classification on them.

Face recognition based on the geometric features of a face is probably the most intuitive approach to face recognition. One of the first automated face recognition systems was described in [Kanade73]: marker points (position of eyes, ears, nose, ...) were used to build a feature vector (distance between the points, angle between them, The recognition was performed by calculating the euclidean distance between feature vectors of a probe and reference image. Such a method is robust against changes in illumination by its nature, but has a huge drawback: the accurate registration of the marker points is complicated, even with state of the art algorithms. Some of the latest work on geometric face recognition was carried out in [Bru92]. A 22-dimensional feature vector was used and experiments on large datasets have shown, that geometrical features alone my not carry enough information for face recognition.

The Eigenfaces method described in [TP91] took a holistic approach to face recognition: A facial image is a point from a high-dimensional image space and a lower-dimensional representation is found, where classification becomes easy. The lower-dimensional subspace is found with Principal Component Analysis, which identifies the axes with maximum variance. While this kind of transformation is optimal from a reconstruction standpoint, it doesn't take any class labels into account. Imagine a situation where the variance is generated from external sources, let it be light. The axes with maximum variance do not necessarily contain any discriminative information at all, hence a classification becomes impossible. So, a class-specific projection with a Linear Discriminant Analysis was applied to face recognition in

[BHK97]. The basic idea is to minimize the variance within a class, while maximizing the variance between the classes at the same time.

Recently various methods for a local feature extraction emerged. To avoid the high-dimensionality of the input data only local regions of an image are described, the extracted features are (hopefully) more robust against partial occlusion, illumation and small sample size. Algorithms used for a local feature extraction are Gabor Wavelets ([Wiskott97]), Discrete Cosinus Transform ([Messer06]) and Local Binary Patterns ([AHP04]). It's still an open research question what's the best way to preserve spatial information when applying a local feature extraction, because spatial information is potentially useful information.

2.2 OpenCV

OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.
Adopted all around the world, OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.

OpenCV (Open Source Computer Vision) is a popular computer vision library started by Intel in 1999. The cross-platform library sets its focus on real-time image processing and includes patent-free implementations of the latest computer vision algorithms. In 2008 Willow Garage took over support and OpenCV 2.3.1 now comes with a programming interface to C, C++, Python and Android. OpenCV is released under a BSD license so it is used in academic projects and commercial products alike.

OpenCV 2.4 now comes with the very new FaceRecognizer class for face recognition, so you can start experimenting with face recognition right away. This document is the guide I've wished for, when I was working myself into face recognition. It shows you how to perform face recognition with FaceRecognizer in OpenCV (with full source code listings) and gives you an introduction into the algorithms behind. I'll also show how to create the visualizations you can find in many publications, because a lot of people asked for.

The currently available algorithms are:

- Eigenfaces

- Fisherfaces
- Local Binary Patterns Histograms

2.3 Face Database

Let’s get some data to experiment with first. I don’t want to do a toy example here. We are doing face recognition, so you’ll need some face images! You can either create your own dataset or start with one of the available face databases, http://face-rec.org/databases/ gives you an up-to-date overview. Three interesting databases are (parts of the description are quoted from http://face-rec.org):

- AT&T Facedatabase The AT&T Facedatabase, sometimes also referred to as **ORL Database of Faces**, contains ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

- Yale Facedatabase A, also known as Yalefaces. The AT&T Facedatabase is good for initial tests, but it’s a fairly easy database. The Eigenfaces method already has a 97% recognition rate on it, so you won’t see any great improvements with other algorithms. The Yale Facedatabase A (also known as Yale faces) is a more appropriate dataset for initial experiments, because the recognition problem is harder. The database consists of 15 people (14 male, 1 female) each with 11 grayscale images sized 320x243 pixel. There are changes in the light conditions (centre light, left light, right light), facial expressions (happy, normal, sad, sleepy, surprised, wink) and glasses (glasses, no-glasses). The original images are not cropped and aligned.

- Extended Yale Face database B The Extended Yale Face database B contains 2414 images of 38 different people in its cropped version. The focus of this database is set on extracting features that are robust to illumination, the images have almost no variation in emotion/occlusion/.... I personally think, that this dataset is too large for the experiments I perform in this document. You better use the AT&T Face database for initial testing. A first version of the Yale Face database B was used in [BHK97] to see how the Eigenfaces and Fisher faces method perform under heavy illumination changes. [Lee05] used the same setup to take 16128 images of 28 people. The Extended Yale Face database B is the merge of the two databases, which is now known as Extended Yale face database B.

2.4 Face detection using haar cascades

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. For this, Haar features shown in the below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

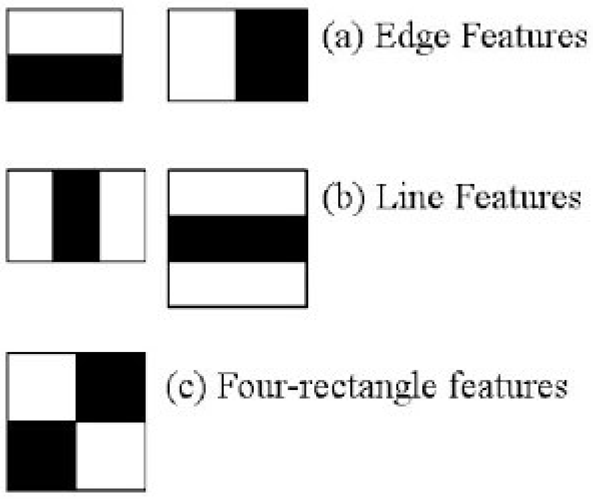


Figure 1: Haar Features

Now, all possible sizes and locations of each kernel are used to calculate lots of features. It would require so much of computation, Even a 24x24 window results over 160000 features). For each feature calculation, we need to find the sum of the pixels under white and black rectangles. To solve this, they introduced the integral image. However large your image, it reduces the calculations for a given pixel to an operation involving just four pixels. Nice, isn't it? It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applied to cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.

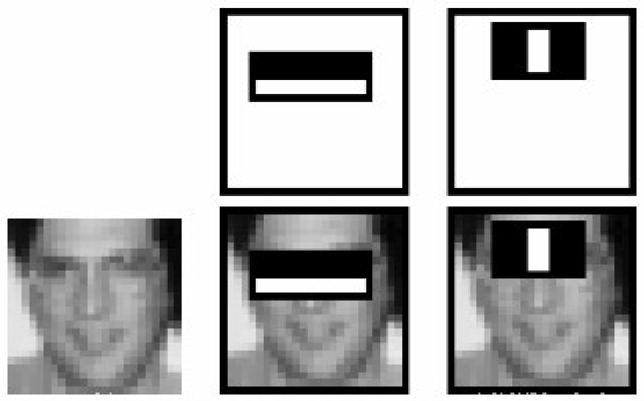


Figure 2: Haar Features

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then the same process is done. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found).

The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

So now you take an image. Take each 24x24 window. Apply 6000 features to it. It is little inefficient and time consuming.

In an image, most of the image is non-face region. So, it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. This way, we spend more time checking possible face regions.

For this they introduced the concept of Cascade of Classifiers. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. (Normally the first few stages will contain very many fewer features). If a window fails the first stage, discard it. We don't consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region. How is that plan!

The authors' detector had 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in the first five stages. (The two features in the above image are actually obtained as the best two features from Adaboost). According to the authors, on average 10 features out of 6000+ are evaluated per sub-window.

So, this is a simple intuitive explanation of how Viola-Jones face detection works. Read the paper for more details or check out the references in the Additional Resources section.

CHAPTER-3
RELATED WORK

CHAPTER-3
RELATED WORK

3.1Literature Review

Development of automated face recognition started in the 1960s, the first semi-automated system for face recognition required the user to locate features (such as eyes, ears, nose and mouth) on the photographs before it calculated distances and ratios to a common reference point, which were then compared to reference data. In the 1970s, Goldstein, Harmon and Lesk used 21 specific subjective markers such as hair color and lip thickness to automate the recognition. The problem with both of these early solutions was that the measurements and locations were manually computed. In 1988, Kirby and Sirovich applied principle component analysis, a standard linear algebra technique, to the face recognition problem. This was considered somewhat of a milestone as it showed that less than one hundred values were required to accurately code a suitably aligned and normalized face image. In eigenfaces techniques, the residual error could be used to detect faces in images, a discovery that enabled reliable real time automated face recognition systems.

The project on face recognition had helped the author to have a detailed survey of a number of face recognition algorithms along with their advantages and limitations. Some of the important methods studied will be described in this section. Face recognition systems architecture broadly consists of the three following tasks:

1. Acquisition (Detection tracking of face-like images)
2. Feature extraction (Segmentation, alignment & normalization of the face image)
3. Recognition

3.2 Drawback of existing systems

1. Current CCTV cameras, security cams are just not enough for security because they had to be monitored by human and a human eye can only focus on two, three faces at a time on monitor leaving other faces unmonitored it is very difficult to find a suspect’s face in the crowded area through the camera on the monitor screen.

2. Cameras enable users to record footage for later viewing, and to help nab criminals, and receive justice from the law. They cannot, however, stop a crime when it is in progress. They do not alert neighbours or the police like an alarm system would. This means that you will incur losses even as you run to the court, make insurance claims and reorder stolen inventory, which may no longer make you feel absolutely safe and even cause you to lose faith in them. People believe that a high number of cameras results in less crime. However, we know that CCTV cameras can record the mishappenings but can't prevent or alert an undesirable event.
3. The cost of the license for an efficient commercial Face recognition system is very high which shows the significant value of the problem. An automatic security system using facial biometrics would provide the needed solution. The problem of face recognition has gained even more importance after the recent increase in the terrorism related incidents. Use of face recognition for authentication also reduces the need of remembering passwords and can provide a much greater security if face recognition is used in combination with other security measures for access control

3.3 Proposed Solution

An efficient face recognition system can be of great help in forensic sciences, identification for law enforcement, surveillance, authentication for banking and security system, and giving preferential access to authorized users i.e. access control for secured areas etc. This project aims to build an opensource facial recognition system which has wide implementation in security and surveillance and minimizing man power by using computation for security as much as possible. This application can be installed on the monitoring desktop taking input for cameras like (webcams, CCTVs, security camps, IP camera,) detecting face real time Identifying it by matching images from the database and the same time save record with details like person name, place where he/she was seen and at what time, etc. Here, the application is using java run time environment, java servlet, OpenCV library Using this software, system perform multiple face detection, recognition, tracking suspect location and time by updating the log.
This technology supports the modern trends of face detection and recognition technology and easily accessible, more secure

We propose a real-time face recognition system based on Viola-Jones algorithm (Haar Cascade Classifier) and PCA.
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.

The main challenge for a face recognition system is of effective feature extraction. The proposed system utilizes the Eigen face method is information reduction for the images. There is an incredible amount of information present even in a small face image. A method must be able to break down pictures so as to effectively represent face images rather than images in general. ‘base faces are generated and then image being analyzed can be represented by the system as a linear combination of these base faces. Each face that we wish to classify can be projected into face-space and then analyzed as a vector. A k-nearest-neighbor approach, a neural network or even a simple Eucledian distance measure can be used for classification. The proposed system uses Principal Component analysis for feature extraction and various distance classifiers the technique used here involves generating the ‘eigen faces’ then projecting training data into face-space to be used with a predetermined classification method and evaluation of a projected test element by projecting it into face space and comparing to training data.

Further increase in functionality can be to send notification to owner, authority, neighbors or triggering alarm for intrusion, if it’s any unknown, criminal or any convicted person and that could lead to stopping crime before it actually happens.

CHAPTER-4
SYSTEM ENGINEERING

CHAPTER-4
SYSTEM ENGINEERING

4.1 System Engineering

Open CV.0

It stands for Open Source Computer Vision; it was designed especially for computational efficiency with strong focus on real time applications. It is written in optimized C/C++, and can take advantage of multi-core processing. In Image processing it has been a great boon for the developers.

Open CV Libraries for java

1. OpenCV for java is a cross platform java wrapper to the OpenCV image processing library. Allowing OpenCV functions to be called from java program. The wrapper can be compiled and run on Windows and Linux.
2. Add the opencv-3xx.jar as external jar in eclipse user libraries and choose “opencv\build\java\x64” directory as “native library location” files required for OpenCV functions to work in project.

Camera captures application

The in-build camera (Desktop Application) is used to capture the images by following steps,
1. Desktop camera is ON, capturing images continuously
2. An Image should be displayed in a java servlet Image box
3. The application should start when “Start” button is pressed and pause when it is again pressed and vice versa.

Face detection

Face detection is a computer vision technology that determines the locations and sizes of human faces in arbitrary digital images. It detects facial features and ignores anything else, such as buildings, trees, background and bodies. Face detection can be considered as object class detection. In object-class detection, the task is to find the locations and sizes of all objects in a digital image that belong to a given class. There are many methods to detect a face in a real-time application. Some ways are easier and some are harder. Face detection approaches used in the project is the Viola Jones method model for Face detection. ‘Haar Cascade Classifier’ is used for detect faces from live desktop camera.

Extract the detected faces from input image

Each extracted face added to the Extracted Faces array. Then display each extracted face in the array to the picture box. Code behind the added buttons to navigate through the extracted faces array to display previous or next extracted face.

Face recognition

"Face recognition is the task of identifying an already detected object as a KNOWN or UNKNOWN face, and in more advanced cases, telling EXACTLY WHO'S face it is!" Face detection is to identify an object as a "face" and locate it in the input image. Face Recognition nothing but is to decide if this "face" is someone KNOWN, or UNKNOWN, basing on the database of faces it uses to validate this input face. PCA based Eigenface method is at the most primary level and simplest of efficient face recognition algorithms

4.2 Requirement Analysis

4.2.1 Hardware Requirement

1. Initially, the algorithm needs high end server to training a cascade classifier.
2. Webcam, camera, IP cameras etc
3. Desktop with
    1. Intel® Core™ i5 Processors or above
    2. 6 GB Ram or above
    3. Webcam, camera, IP cameras etc.

4.2.2 Software Requirements

1. NetBeans IDE: NetBeans is an integrated development environment (IDE) for Java. NetBeans allows applications to be developed from a set of modular software components called modules. NetBeans runs on Microsoft Windows, macOS, Linux and Solaris. In addition to Java development, it has extensions for other languages like PHP, C, C++, HTML5, Javadoc, and JavaScript. Applications based on NetBeans, including the NetBeans IDE, can be extended by third party developers.

The NetBeans Team actively supports the product and seeks feature suggestions from the wider community. Every release is preceded by a time for community testing and feedback.

2. Eclipse IDE Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java IDE. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications

   The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles

   Eclipse software development kit (SDK) is free and open-source software, released under the terms of the Eclipse Public License, although it is incompatible with the GNU General Public License. It was one of the first IDEs to run under GNU Class path and it runs without problems under IcedTea.
3. JDK (Java Development Kit) to build and compile the java servlet application Java Development Kit (JDK) is a superset of a JRE and contains tools for Java programmers, e.g. a javac compiler. The Java Development Kit is provided free of charge either by Oracle Corporation directly, or by the OpenJDK open source project, which is governed by Oracle.
4. A Java virtual machine (JVM) is an abstract computing machine that enables a computer to run a Java program. There are three notions of the JVM: specification, implementation, and instance. The specification is a document that formally describes what is required of a JVM implementation. Having a single specification ensures all implementations are interoperable. A JVM implementation is a computer program that meets the requirements of the JVM specification. An instance of a JVM is an implementation running in a process that executes a computer program compiled into Java bytecode.
5. JRE (Java Runtime Environment) to execute the compiled java application.

Java Runtime Environment (JRE) is a software package that contains what is required to run a Java program. It includes a Java Virtual Machine implementation together with an implementation of the Java Class Library. The Oracle Corporation, which owns the Java trademark, distributes a Java Runtime environment with their Java Virtual Machine called Hotspot.
OpenCV Library (opencv-3xx.jar, opencv_java3xx.dll), OpenCV 2.4.4, OpenCV supports desktop Java development

4.2.3 Functional Requirements

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 to training a cascade classifier.
The classifier trained using these data sets can used to recognize face in images, videos or real time stream from cameras the classifier trained using frontal face images are haarcascades, lbpcascades.
These cascade classifiers are used to detect faces, crop and save as new images after that a new model is trained using the cropped face to finally recognize faces in the frames of video image or camera stream.

4.3 System Analysis

4.3.1 Planning and Scheduling

The Software Development Plan is a comprehensive, composite artifact that gathers all information required to manage the project. It encloses a number of artifacts developed during the Inception phase and is maintained throughout the project.
The project was just an idea planned in January then before starting the project problem statement was decide why this project is needed ? to be implemented and what is the drawback of current system ? and what drawback this projects can eliminate ? then requirements are gathered like hardware, software, sensors that is camera or webcam after collecting the requirements design interface is built in February, after complete interface backend implemented. Backend implementation was completed in May 2018 after that adding new features and training, testing was completed on 05 June 2018.

4.3.2 Cost Estimation

Estimated cost is zero because all the software's used are open source or free to use and no extra hardware is required other than the laptop with the webcam.

4.3.3 Feasibility Study

The feasibility analysis will be performed to determine whether the system is capable of performing the intended job. The pilot project will be executed, and if successfully implemented and achieved desired output from the system.
For that the final application was first tested on computer system with configuration as follows:

Computer System 1 :

Processor	:	Intel® Core™ i3 Processors
Ram	:	4 GB Ram
Camera Sensor	:	Webcam, camera, IP cameras etc.

Results	:	Application lagged because of less processing power and RAM.

Second time application was implemented on computer system with better configuration

Computer System 2:

Processor	:	Intel® Core™ i5 Processors
Ram	:	6 GB Ram
Camera Sensor	:	Webcam, camera, IP cameras etc.

Results	:	Application run successfully within given polynomial time faces were detected, tracked, recognized successfully, security is improved.
Conclusion	:	Project is totally feasible and the system is capable of performing the intended job

4.5 Analysis Modeling

4.5.1 Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).
A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show information about process timing or whether processes will operate in

sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data flows as a unified model.

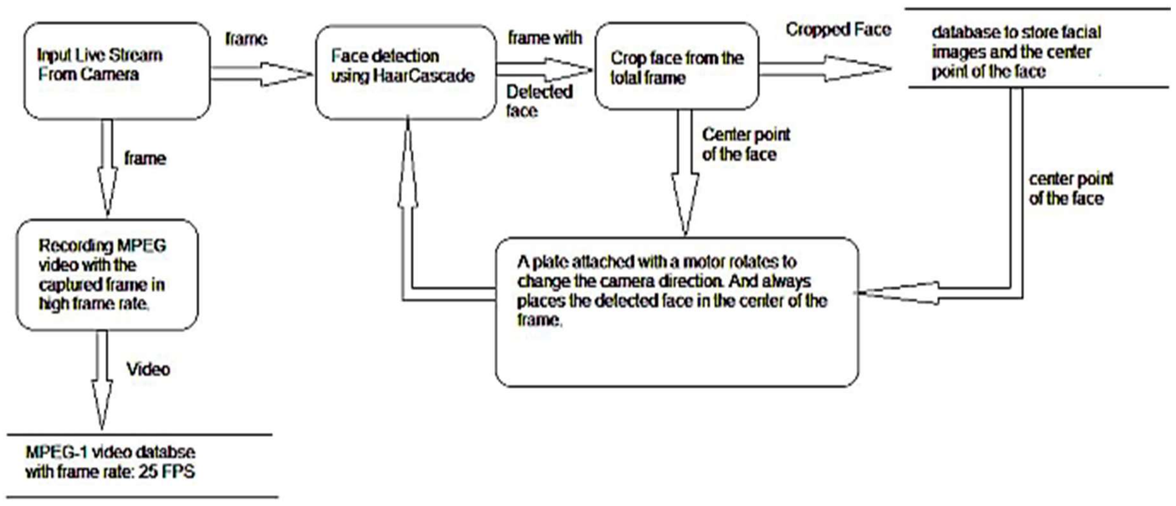

Input Live Stream From Camera
frame
Face detection using HaarCascade
frame with Detected face
Crop face from the total frame
Cropped Face
database to store facial images and the center point of the face
frame
Center point of the face
center point of the face
Recording MPEG video with the captured frame in high frame rate.
A plate attached with a motor rotates to change the camera direction. And always places the detected face in the center of the frame.
Video
MPEG-1 video databse with frame rate: 25 FPS

Figure 3: Data Flow Diagram

CHAPTER-5
SYSTEM MODELING

CHAPTER-5
SYSTEM MODELING

5.1 Use case Diagram

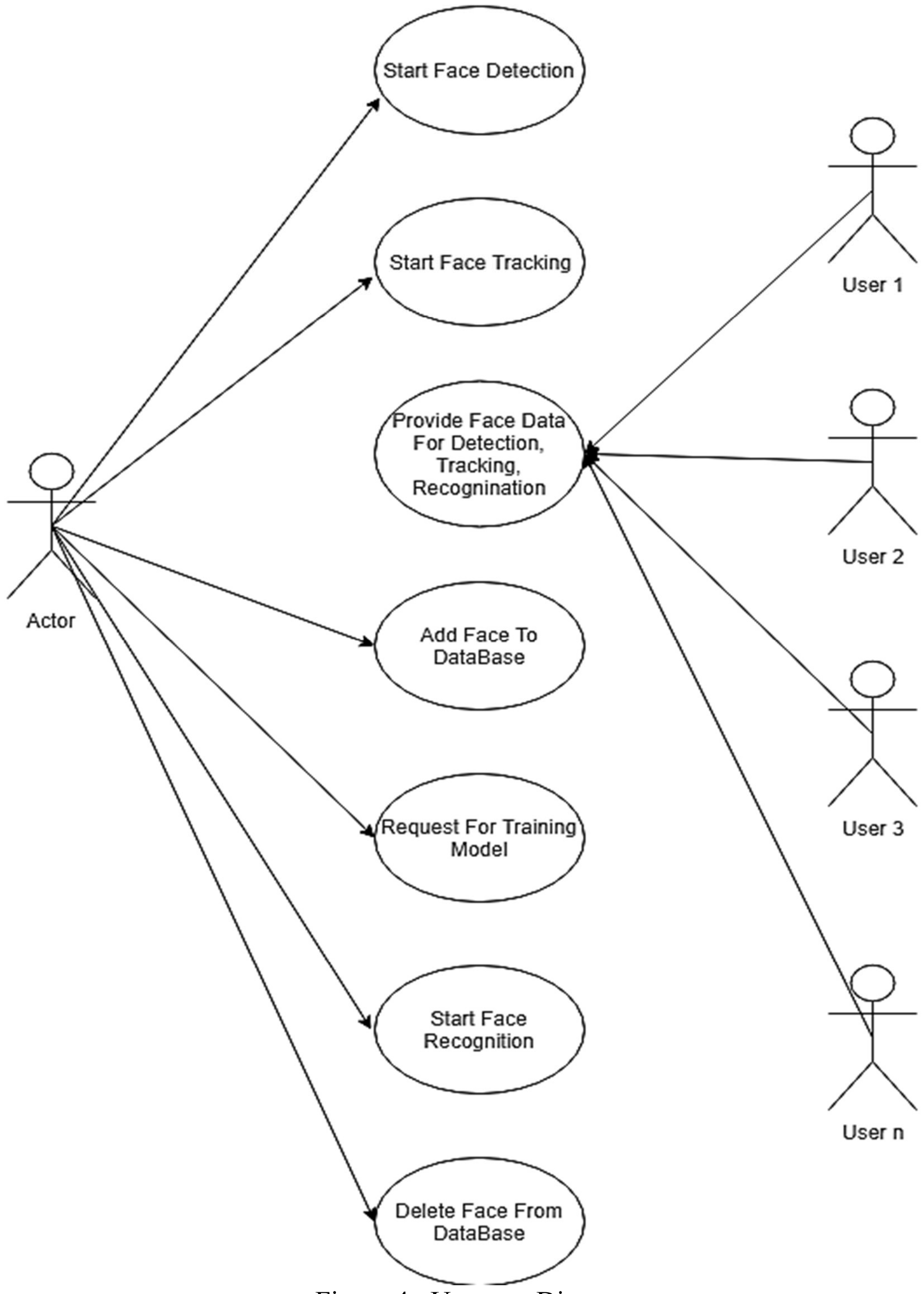


Figure 4 : Use case Diagram

5.1 Activity Diagram

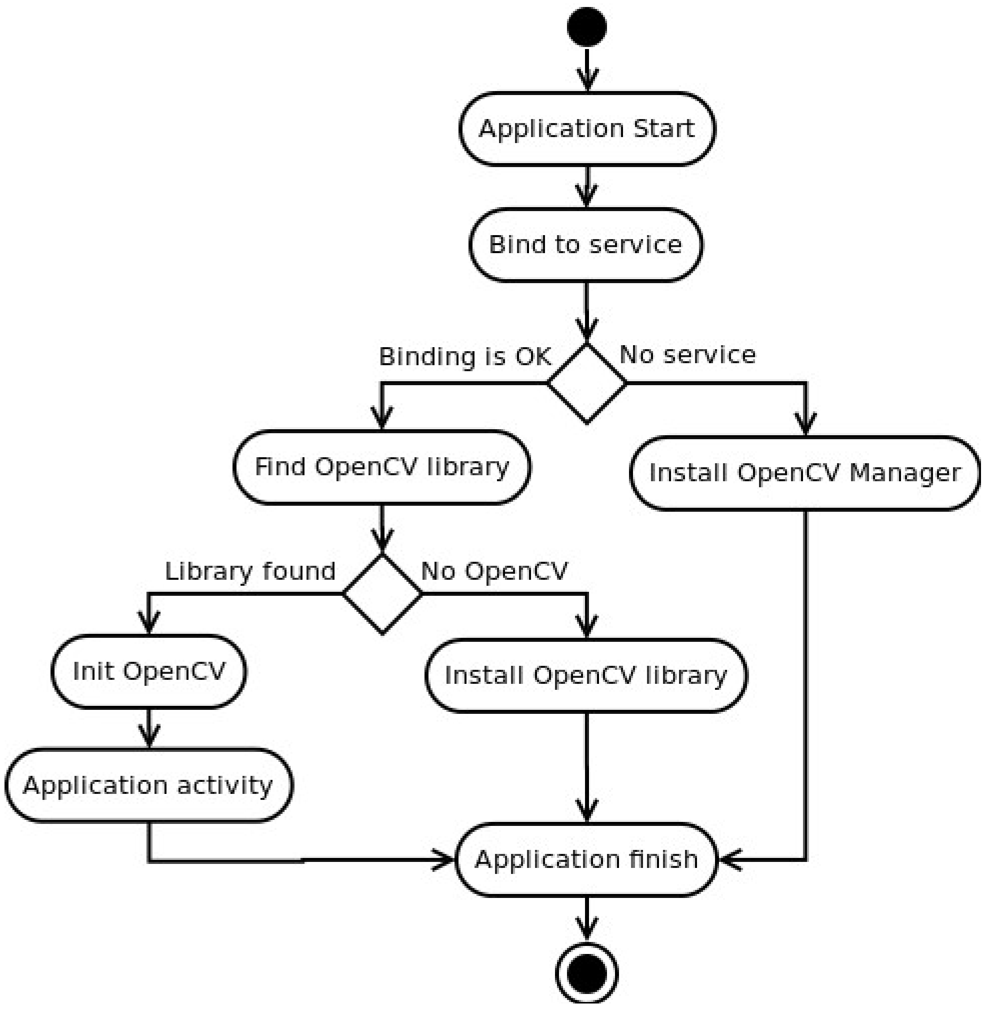


Figure 5: Activity Diagram

5.1 Class Diagram

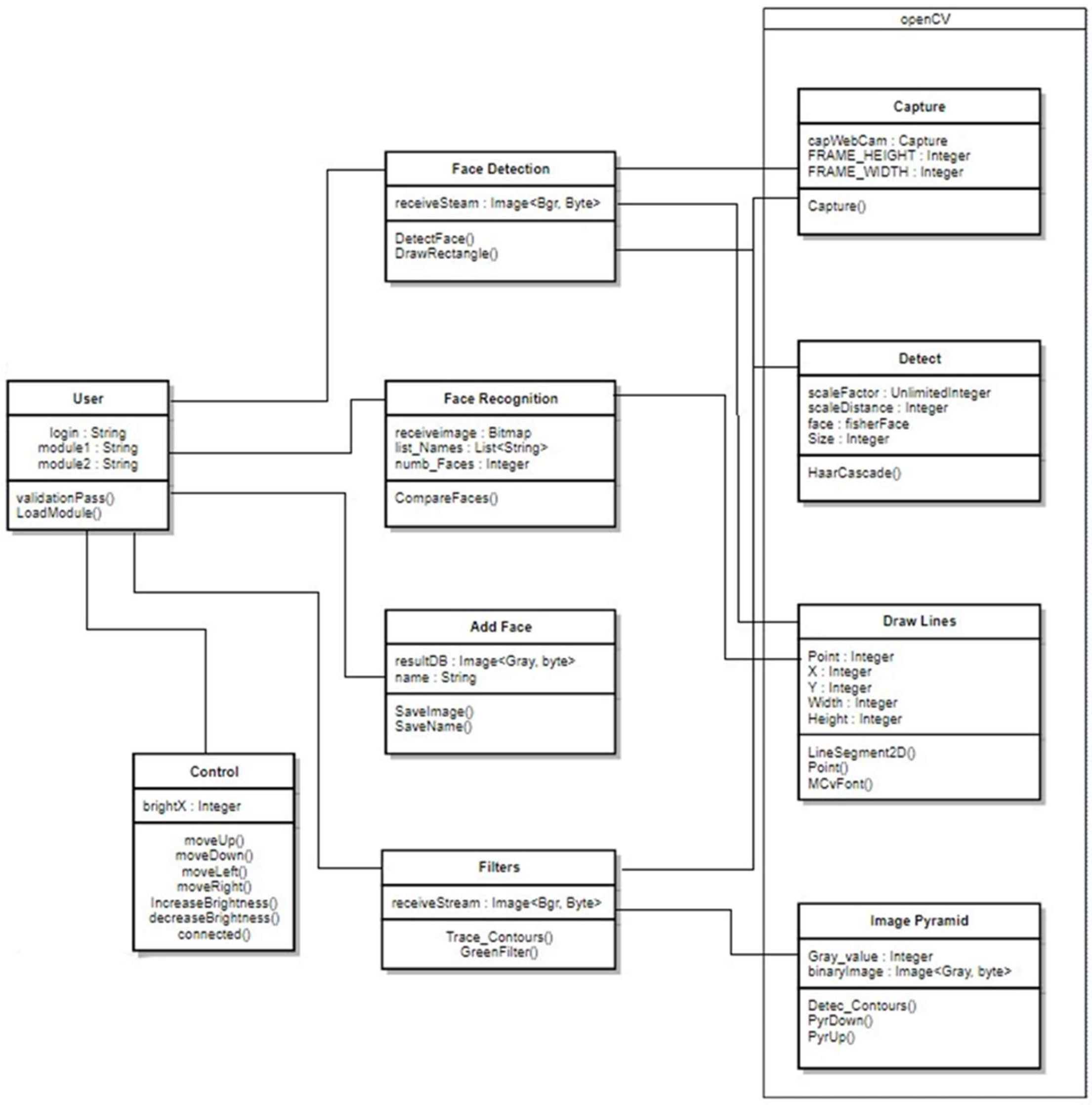


Figure 6: Class Diagram

5.1 Sequence Diagram

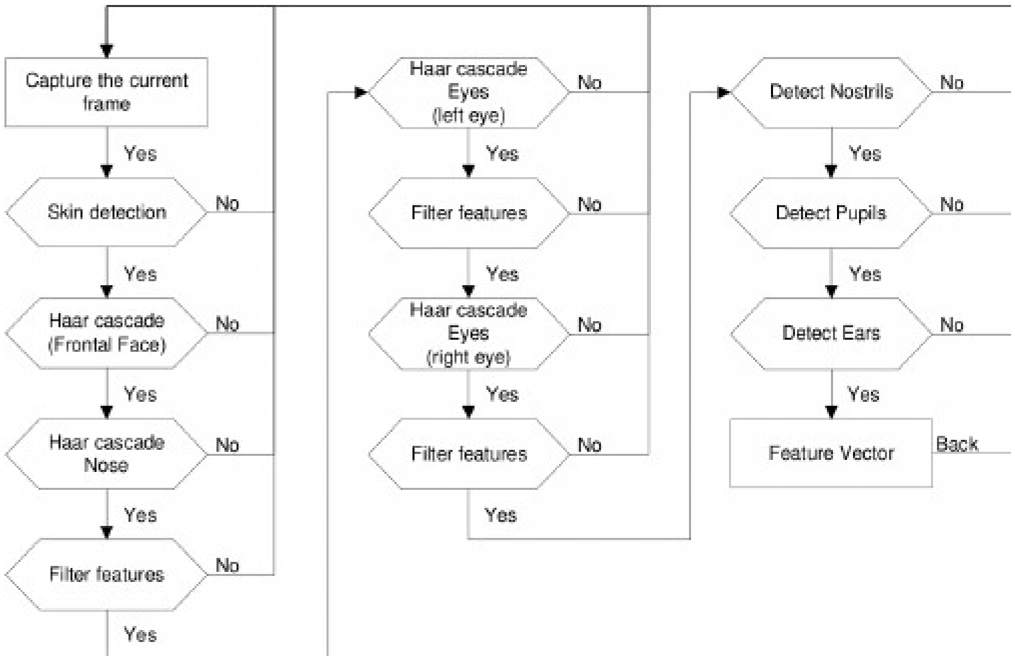


Figure 7: Sequence Diagram

5.1 Deployment Diagram

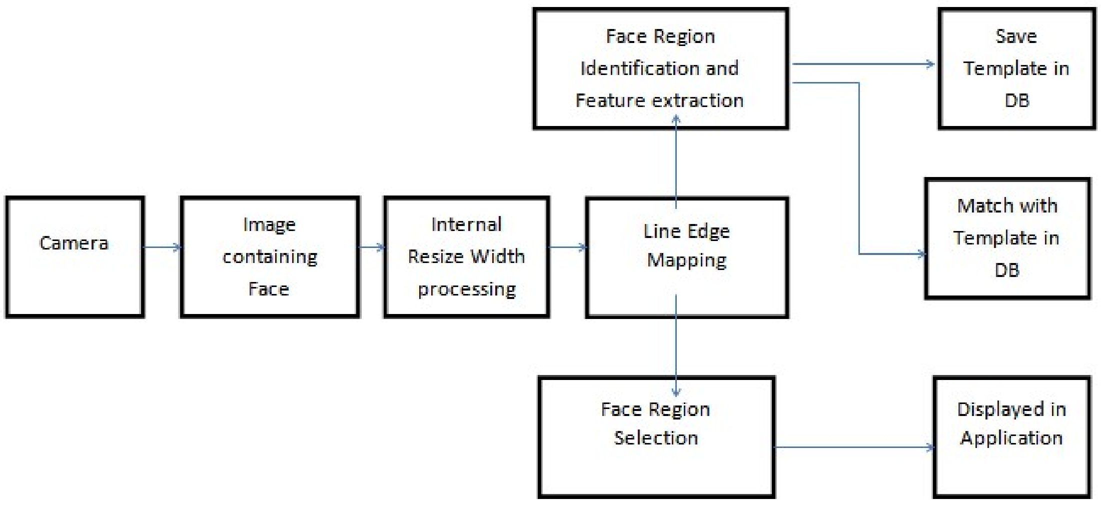


Figure 8: Deployment Diagram

CHAPTER-6
SYSTEM DESIGN

CHAPTER-6
SYSTEM DESIGN

6.1 Design Overview

6.1.1 System Architecture

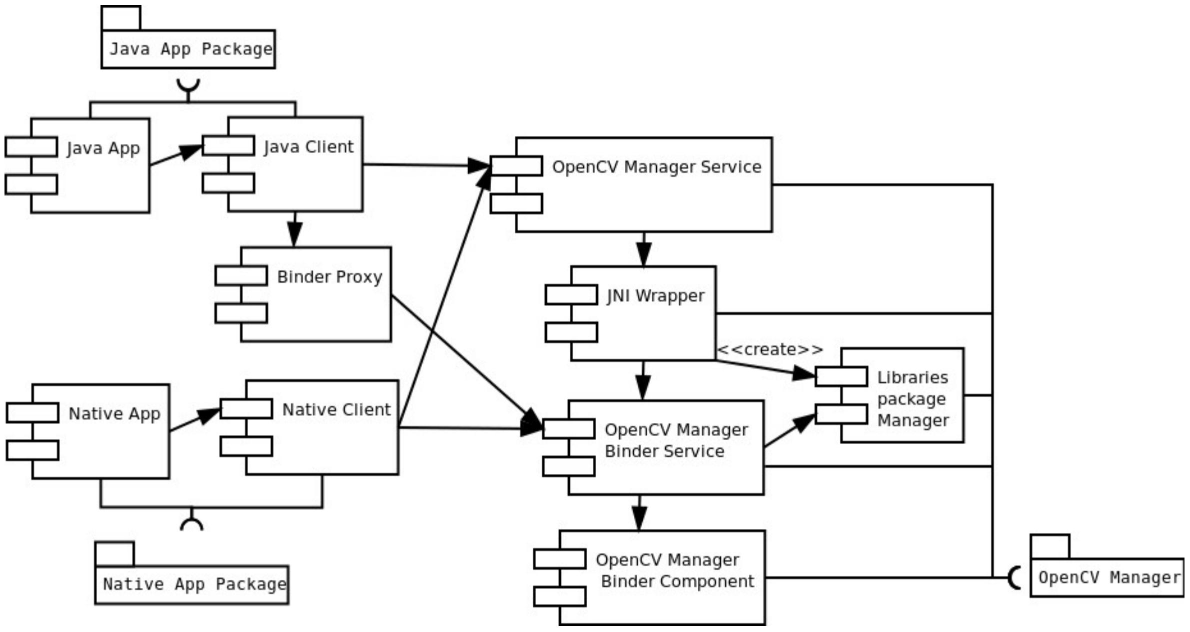


Figure 9: Application Architecture

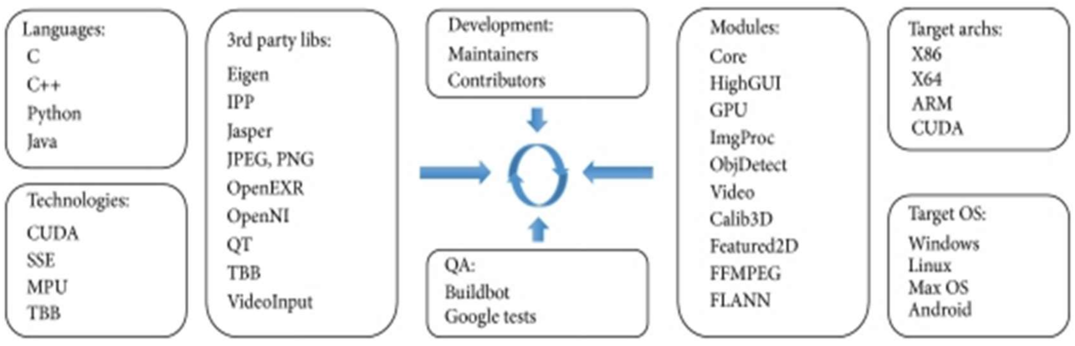


Figure 10: OpenCV Architecture

CHAPTER-7
SYSTEM IMPLEMENTATION

CHAPTER-7

SYSTEM IMPLEMENTATION

7.1 Implementation

This application is implemented in java and the interface is design using JavaFX, JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc..

JavaFX application code can reference APIs from any Java library. For example, JavaFX applications can use Java API libraries to access native system capabilities and connect to server-based middleware applications.

It is a set of graphics and media packages that enables developers to design, create, test, debug, and deploy rich client applications that operate consistently across diverse platforms.

This application can be installed and implemented on cross platform monitoring desktop taking input from cameras like (webcams, CCTVs, security camps, IP cameras,) detecting face real time.
Identifying it by matching images from the database and the same time save record with details like person name, place where he/she was seen and at what time, etc. Here, the application is using java run time environment, java servlet, jfoenix OpenCV library, haarcascades, lbpcascades Using this software, system perform multiple face detection, recognition, tracking suspect location and time by updating the log.

7.2 User Interface

As sated earlier the application is implemented in java and the user interface is designed in JavaFx, below images show the interface of the application.

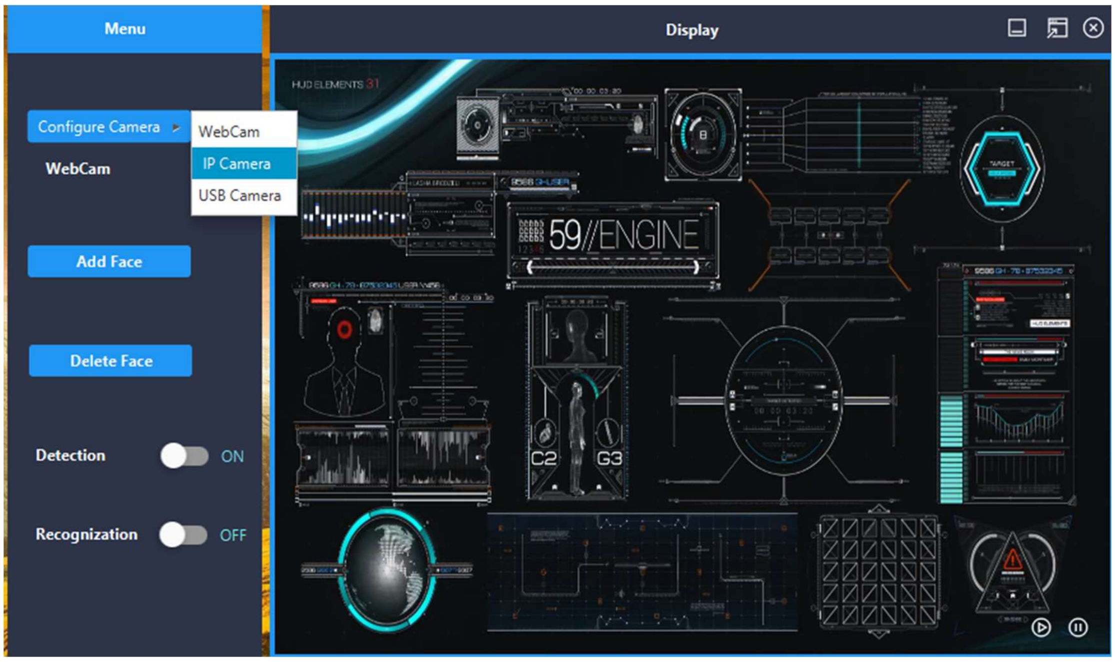


Figure 11: Application launch screen

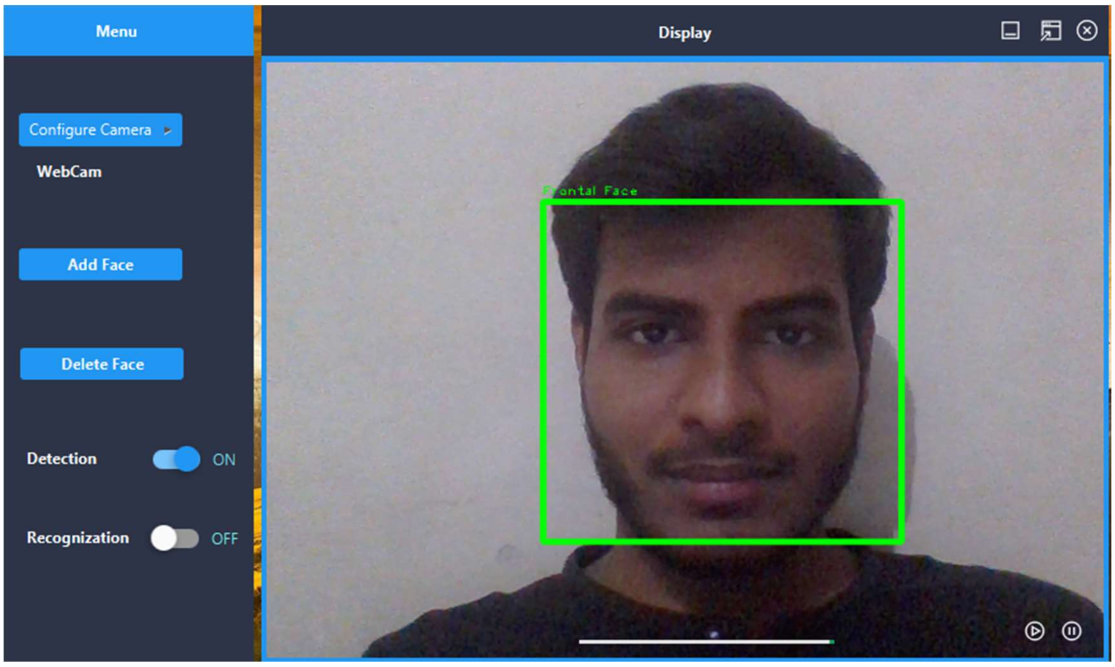
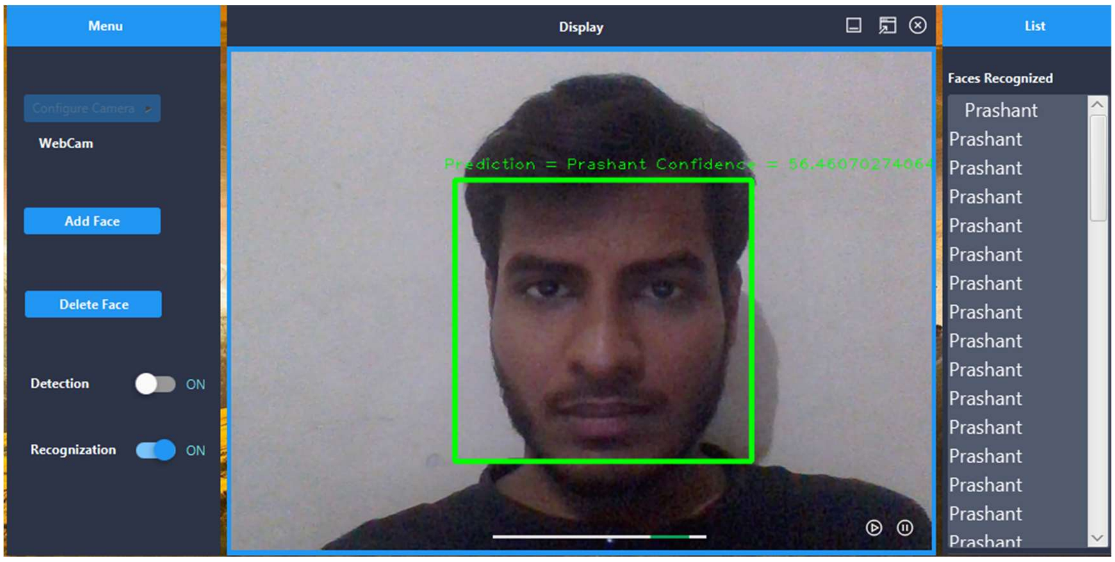


Figure 12: Face detection window

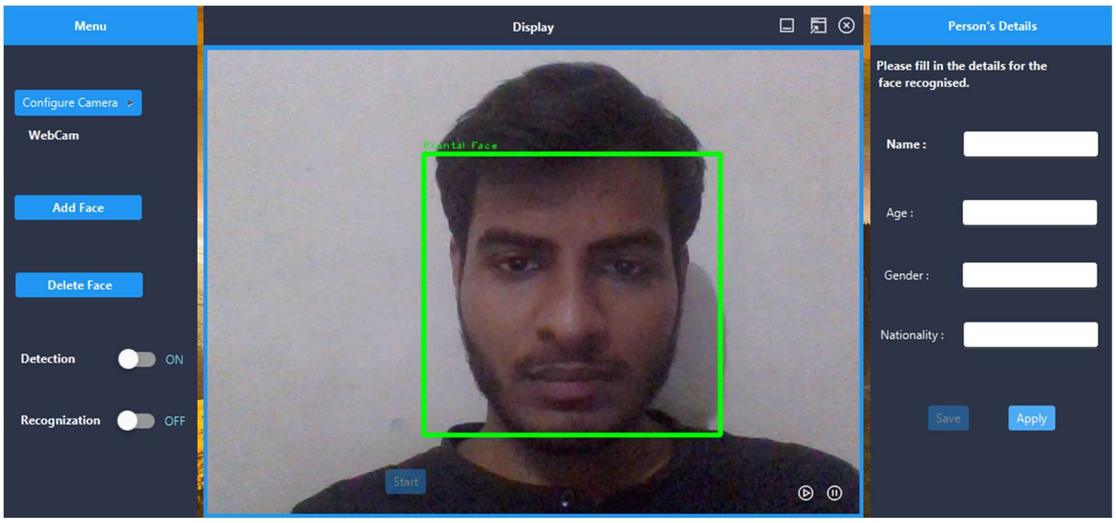


Detection ON
Recognization ON

Prediction = Prashant Confidence = 56.4607027406

Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant
Prashant

Figure 13: Face recognition window



Detection ON
Recognization OFF

Frontal Face

Please fill in the details for the face recognised.

Name :
Age :
Gender :
Nationality :

Figure 14: Adding face into database window

CHAPTER-8
TESTING APPROACH

CHAPTER-8
TESTING APPROACH

8.1 Introduction

Testing is the process of identifying the correctness and quality of software program. The purpose is to check whether the software satisfies the specific requirements, needs and expectations of the customer. In other words, testing is executing a system or application in order to find software bugs, defects or errors. The job of testing is to find out the reasons of application failures so that they can be corrected according to requirements

8.2 Testing Techniques

Software testing techniques listed here are the major methods used while conducting various Software Testing Types during various Software Testing Levels.

Black Box : A software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. Test design techniques include Equivalence partitioning, Boundary Value Analysis, Cause-Effect Graphing.

White Box : A software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. Test design techniques include Control flow testing, Data flow testing, Branch testing, Path testing.

8.2.1 Black Box Testing

BLACK BOX TESTING, also known as Behavioural Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional

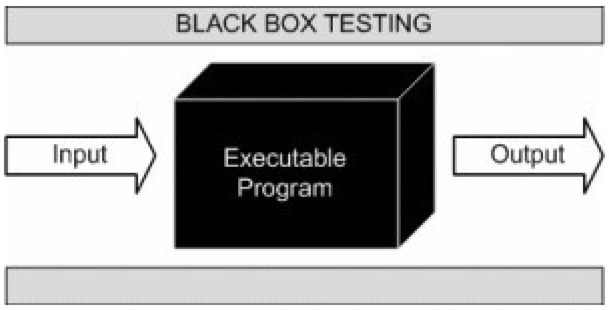


Figure 15: Black Box Testing

This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see.

In this project, we have performed testing on detection, tracking and recognition. Test accuracies and test cases are as follows:

Person	**Confidence**
Prashant	72.5%
Anubhav	80.9%
Sidharth	85.1%
Anshika	90.2%

Table 1 : Confidence level of Face Recognition

8.3 Test Cases

8.3.1 Test Cases of Face Recognition

S No.	**Test case**	**Input** Real Time	**Expected output** Real Time	**Obtained output** Real Time	**Remark**
1	Prashant	Camera Stream	Label: Prashant	Label: Prashant	Pass
2	Anubhav	Camera Stream	Label: Anubhav	Label: Anubhav	Pass
3	Anshika	Camera Stream	Label: Anshika	Label: Anshika	Pass
4	Sidharth	Camera Stream	Label: Sidharth	Label: Sidharth	Pass

5	Prashant, Anubhav	Camera Stream	Label: Prashant Anubhav	Label: Prashant Anubhav	Pass
6	Anshika, Sidharth, Anubhav	Camera Stream	Label: Anshika Sidharth Anubhav	Label: Anshika Sidharth Anubhav	Pass

Table 2 : Test Cases Result of Face Recognition

CHAPTER-9
RESULT ANALYSIS

CHAPTER-9
RESULT ANALYSIS

9.1 Result analysis

9.1.1 Experimental result of face detection :

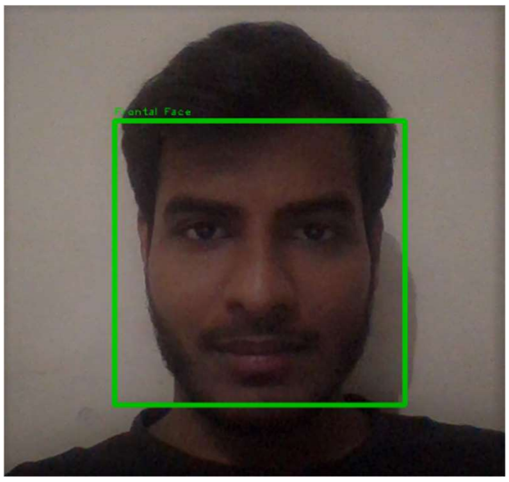


Figure 16: Result of face detection

Results of face detection successfully shows detected face in real time by taking input for the camera stream, application even detects that the face is a front profile face.

9.1.2 Experimental results of face recognition :

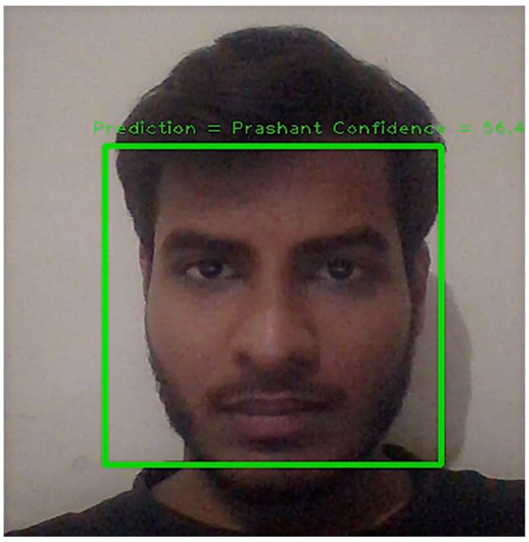


Figure 17: Result of face recognition

Result of face recognition in real time by taking input for the camera stream, successfully show the expected name of the persons face with a confidence level 56.4.Confidence level in face recognition can be increased by increase in quality of camera sensor and training data in good lightning condition.

9.1.3 Experimental result of adding face to the data base :

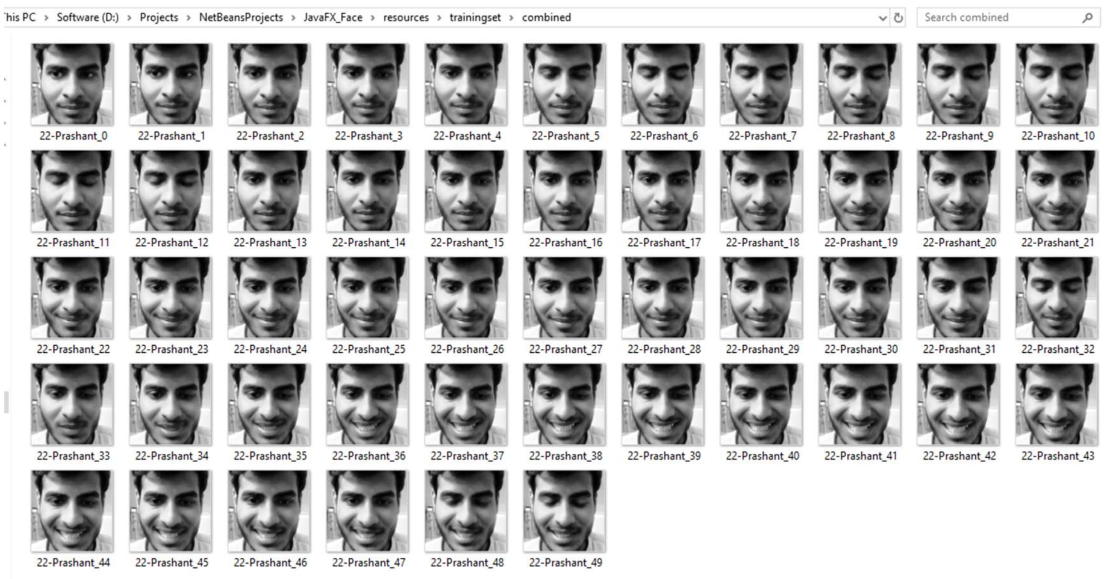


Figure 18: Figure 16: Result of adding face to database

Application successfully detects, crops, resize face image and saves it into file system for training data set for facial recognition

CONCLUSION

Cameras enable users to record footage for later viewing, and to help nab criminals, and receive justice from the law. They cannot, however, stop a crime when it is in progress. They do not alert neighbors or the police like an alarm system would. People believe that a high number of cameras results in less crime. However, we know that CCTV cameras can record the mishappenings but can't prevent or alert an undesirable event.

So to increase security this application system can be implemented with security cameras and CCTV as we have seen positive results and high feasibility through this report.
Face recognition which is implemented in real-time helps to recognize the human faces can be used for person identification and authentication purposes.

This technology can used by Law enforcement agencies for monitoring, finding suspects and criminals on road traffic, railway stations, scan faces in CCTV footage, Border control deployments and can also be used as consumer application in schools, colleges, and corporate offices for attendances and security.

Further implementation can be to send notification to owner, authority, neighbors or triggering alarm for intrusion, if it's any unknown, criminal or any convicted person and that could lead to stopping crime before it actually happens.

This system has unlimited scopes as it can be implement in public places like Railway Stations, Airports, Sea Ports, and Market Places stopping criminal and terrorist suspects from causing undesirable event. It can be implemented in homes, shops, or any personal properties preventing trespassers or troublesome people from causing any harm, damage or burglary by triggering alarm or notification.

FUTURE WORK

1. In future this application can be installed on high end server for better performance which will result in increase of confidence and accuracy in facial detection and recognition.
2. This system can be connected to the internet so that consumers an track and get notified for the events like intrusion, trespassing, or recognitions of some criminal or suspect figure whose face is added to the database earlier.
3. By connecting the application to the internet Admin can allow or authorise person to enter house or any place which requires facial recognition to enter but the persons face is not in the data base so Admin can get notified for entry request and can allow remotely.

APPENDIX

I. Glossary

Sr.No.	Term	Definition
1	PCA	Principal component analysis
2	CCTV	Closed Circuit Television

II. Abbreviations

Sr.No	Acronym	Term
1	PCA	Principal component analysis
2	CCTV	Closed Circuit Television

III. List of figures

IV. List of tables

BIBLIOGRAPHY

References

Research/journal papers:

[ISSN 2105–1232 c 2014]	Yi-Qing Wang An Analysis of the Viola-Jones Face Detection Algorithm Published in Image Processing on Line on 2014–06–26. Submitted on 2013–08–31, accepted on 2014–05–09. ISSN 2105–1232 c 2014 IPOL & the authors CC–BY–NC–SA
[ISSN: 2321 – 2403, 2013]	S.V. Viraktamath, Mukund Katti, Aditya Khatawkar & Pavan Kulkarni. The SIJ Transactions on Computer Networks & Communication Engineering (CNCE), Vol. 1, No. 3, July-August 2013
[Paul Viola 2001]	Paul Viola, Michael Jones Robust Real-time Object Detection (2001) SECOND INTERNATIONAL WORKSHOP ON STATISTICAL AND COMPUTATIONAL THEORIES OF VISION –MODELING, LEARNING, COMPUTING, AND SAMPLING VANCOUVER, CANADA, JULY 13, 2001

PUBLICATIONS

List of publications

Sr. No.	Title of Paper	Name of Authors	Name of the Journal/Publication/conference	Issue/volume/ Date	Impact Factor
1	Face Recognition - A Tool for Automated Attendance System	Prashant			