CHAPTER 1

INTRODUCTION

In computer science, digital image processing is the use of computer algorithms to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems.

1.1 Background

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analysing patterns based on the person's facial textures and shape.

Positive identification of individuals is a very basic societal requirement. In small tribes and villages, everyone knew and recognized everyone else. You could easily detect a stranger or identify a potential breach of security. In today's larger, more complex society, it isn't that simple. In fact, as more interactions take place electronically, it becomes even more important to have an electronic verification of a person's identity.

The problem is, these forms of electronic identification aren't very secure, because they can be given away, taken away, or lost and motivated people have found ways to forge or circumvent these credentials. So, the ultimate form of electronic verification of a person's identity is biometrics; using a physical attribute of the person to make a positive identification.

In many applications like the surveillance and monitoring ,say, of a public place, the traditional biometric techniques will fail as for obvious reasons we can not ask everyone to come and put his/her thumb on a slide or something similar. So we need a system which is similar to the human eye in some sense to identify a person. To cater this need and using the observations of human psychophysics, face recognition as a field emerged.

1.2 Existing Systems and their Drawbacks

Face recognition methods are fall into two main categories: feature-based and appearance based methods. An overview of some of the well-known methods in these categories is given below.

1.2.1 Feature-based approaches

Feature-based approaches first process the input image to identify and extract (and measure) distinctive facial features such as the eyes, mouth, nose, etc., as well as other fiducial marks, and then compute the geometric relationships among those facial points, thus reducing the input facial image to a vector of geometric features. Standard statistical pattern recognition techniques are then employed to match faces using these measurements. Early work carried out on automated face recognition was mostly based on these techniques. One of the well-known feature-based approach is the elastic bunch graph matching method.

The main advantage offered by the featured-based techniques is that since the extraction of the feature points precedes the analysis done for matching the image to that of a known individual, such methods are relatively robust to position variations in the input image. In principle, feature-based schemes can be made invariant to size, orientation and/or lighting. Other benefits of these schemes include the compactness of representation of the face images and high speed matching .

The major disadvantage of these approaches is the difficulty of automatic feature detection and the fact that the implementer of any of these techniques has to make arbitrary decisions about which features are important.

1.2.2 Appearance-based Methods

Appearance based methods approaches attempt to identify faces using global representations, i.e., descriptions based on the entire image rather than on local features of the face. Techniques such as Eigen faces, Principal component analysis (PCA) , linear discriminate analysis (LDA) and independent component analysis (ICA) have demonstrated the power of appearance based methods both in ease of implementation and in accuracy. Here the feature vector used for classification is a linear projection of the face image into a low-dimensional linear subspace. In extreme cases, the feature vector is chosen as the entire image, with each element in feature vector taken from a pixel in the image.

The main advantage of the Appearance-based approaches is that they do not destroy any of the information in the images by concentrating on only limited regions or points of interest. However, as mentioned above, this same property is their greatest drawback, too, since most of these approaches start out with the basic assumption that all the pixels in the image are equally important.

Perfecting face recognition technology is dependent on being able to analyze multiple variables, including lighting, image resolution, uncontrolled illumination environments, scale, orientation (in-plane rotation), pose (out-of-plane rotation), people's expressions and gestures, aging, and occlusion (partial hiding of features by clothing, shadows, obstructions, etc.).This is highly challenging for computer scientists. Solutions are largely mathematical, with new procedural and machine-learning algorithms being developed to improve accuracy.

1.3 Proposed System

The Real-Time Person Location system will provide a Web-based UI to input the name of a person. A camera interfaced with the proposed application will be streaming live footage of an environment. Location tags will be made visible in this live stream. When the Deep Learning software recognizes the desired person, it will return the location of the person in question, from the stored location tags obtained from performing text recognition. It will also return a video frame capturing the person. The user will be provided with an additional option of viewing the names of all the other people that were sitting in the same room, at that same instant the person in question was found.

1.4 Advantages of the proposed system

- The key advantage of a facial recognition system is that it is able to person mass identification as it does not require the cooperation of the test subject to work. Properly designed systems installed in airports, multiplexes, and other public places can identify individuals among the crowd, without passers-by even being aware of the system.
- Reduced missed detections-The proposed system makes less missed detections compared to the existing system.
- Reduced false detections and Duplicate detctions-These early approaches are also susceptible for detecting non human objects as humans. A trade-off between Missed Detections and False Detections can be achieved by adjusting the threshold parameters.

CHAPTER 2

LITERATURE REVIEW

A Literature Survey or Narrative Survey is a type of survey article. A literature survey is a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic.

In [1], the authors talk about the results obtained when creating a prototype of a software complex that implements speech recognition in the video stream by the motion of the lips with the help of a neural network. The construction of a neural network model based on Long Short-Term Memory (LSTM) layers is also described here.

LSTM are units of a Recurrent Neural Network (RNN). An RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. LSTM networks are well suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. The authors explain about the identification of a person uniquely based on the face, shape of the lips and their movements to increase the accuracy of the identification and to reduce the rate of false positive.

In [2], the authors present a way to increase face recognition quality in video surveillance system an approach of censoring incoming photos based on FRiS function. The Function of Rival Similarity or FRiS-function. This function is of relative nature and it is in good agreement with the mechanism of human perception of similarity and difference. There has been a significant improvement in recognition precision with a minimum number of rejections from recognition, namely, the precision has increased from 96% to 99.5% with 10% rejections.

In [3], the authors talk about the core functionality of Real Time Face Tracking and Identification for Surveillance System is to allow tracking and recognition of human faces in a video stream and thereby provide a centralized, cost effective and robust mechanism of securing business and government premises. In this paper, a powerful multiple face recognition system has been proposed. The system is built, using a combination of Histogram of Oriented Gradient (HOG) feature descriptor and Support Vector Machines (SVM) feature classifier along with the Viola-Jones face detection framework. Jones object detection framework is used for face detection and HOG feature descriptor is used for face feature extraction.

In [4], the system proposed by the authors estimated the pose automatically in real-time by utilizing the detected landmarks information, with pose variation face images stored for matching templates. Through the camera or video stream, it captures the face image, as shown in Figure 2.1 and 2.2, uses supervised descent method based facial landmark localization for face detection and fits the curve to estimate the area of left and right face.

An interface prompts "the positive face collection", one look at the center of the camera for face image acquisition for a predefined time; when the interface prompts "the left (or right) face collection ", one look at the left (or right) of the camera for collecting the left or right pose of faces. After that, one will see the "successfully registered" inform, and the overcomplete face registering process is accomplished. The registered images are compared by the recognition algorithm and the images are sorted according to the similarity.

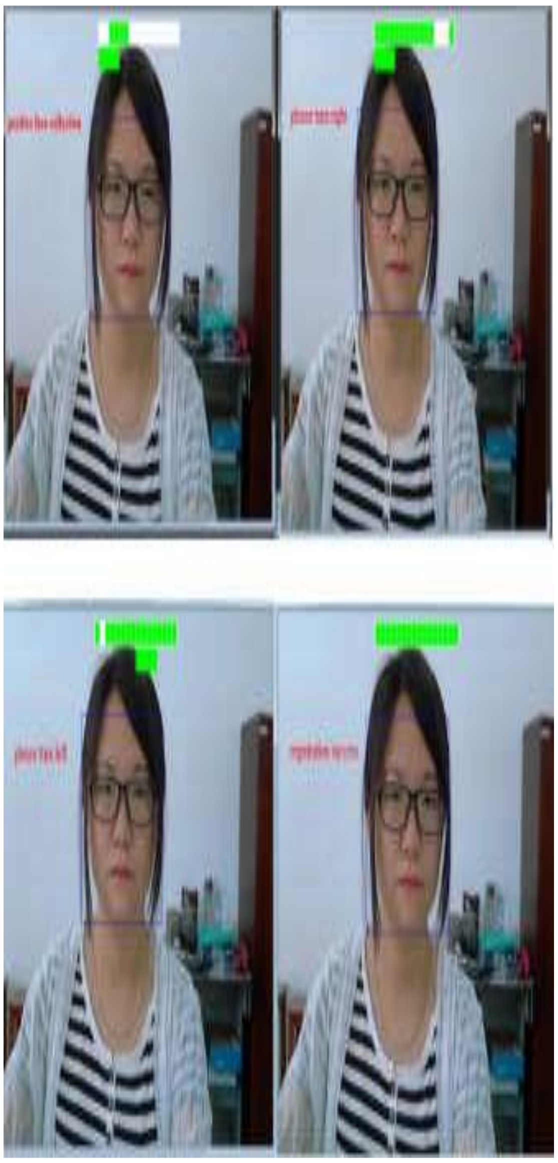


Figure 2.1 Real-time camera registration

Figure 2.2 Registered face images

In [5], the system proposed by the authors identifies and authenticates each student present in the class, as shown in Figure 2.3. Through this system, the attendance of group of students is carried out at a single point of time. Eigen face algorithm is used, taken from the concept of Eigen vectors that is being used in matrix to calculate the matrix in an efficient way. In the same way Eigen is used in face recognition to make this system faster and efficient. When image of a face is taken it gets centered by using the eyes. This algorithm follows a process through which the face is divided into smaller parts called as Eigen vectors, that are used in this process of calculation.

Viola Jones is extensively used for object detection in real time. Its basic requirement is complete frontal images of the face (face must not be tilted or so and should point towards the camera). It is robust in nature since the detection rate is high and has got very low false positive rate. This system consists of a database containing the primary information of the students along with their photographs linked via cloud. The faculty login into the respective account to access the system. If the student is not yet registered, then the faculty carries out the process of registration. During this process the faculty needs to enter the id of the student in order to give each student’s image a unique id i.e. the Enrollment Number of the student. The next step is creation of the data set where the face of the student is captured for 10 seconds at different angles and a data set of 20 images for a single student is automatically created.

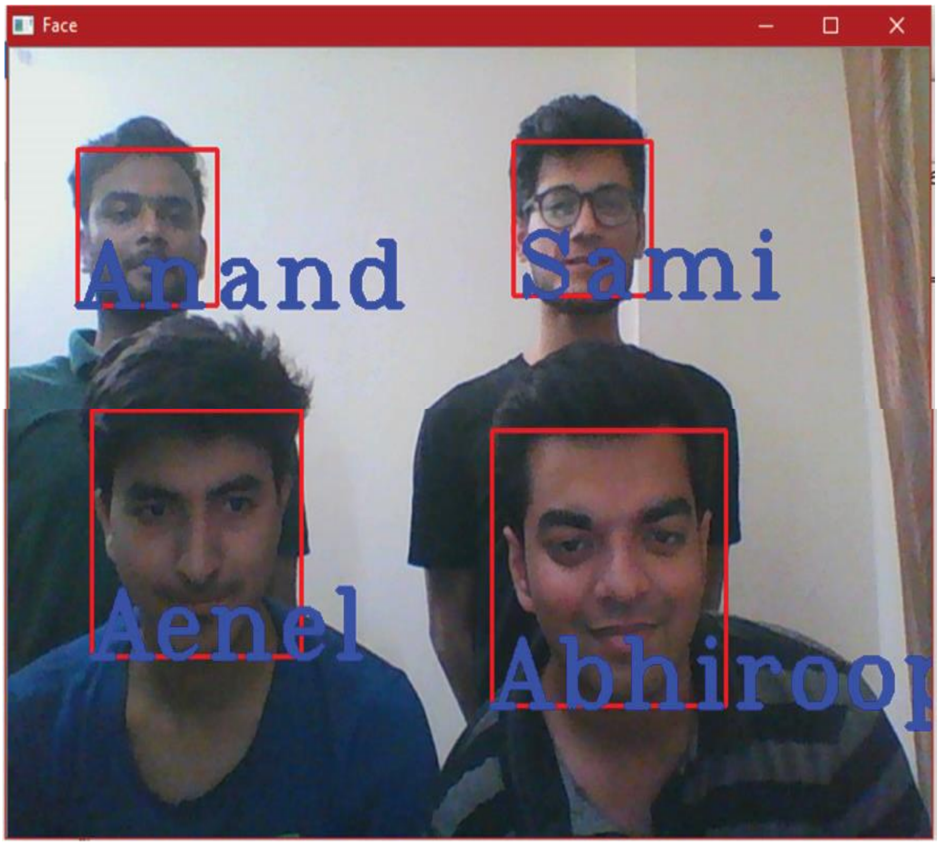


Figure 2.3 Group face Detection and Recognition

In [6], the authors propose a system that recognized human emotions (happiness, sadness, surprise, anger, disgust, neutral or any combination of the 6 emotions), as shown in Figure 2.4, based on facial expressions using a webcam. The deep Convolution Neural Network (CNN) model was developed by using Caffe for real-time application operated on Windows system. It was based on CUDA, so it was faster than CPU. It used a webcam to recognize faces and facial expressions within a distance of 2~3m for TV environment. It recognized 6 primary emotions and 3 secondary emotions. The recognition rate of the personalized facial expression recognition was 96% in 3m.

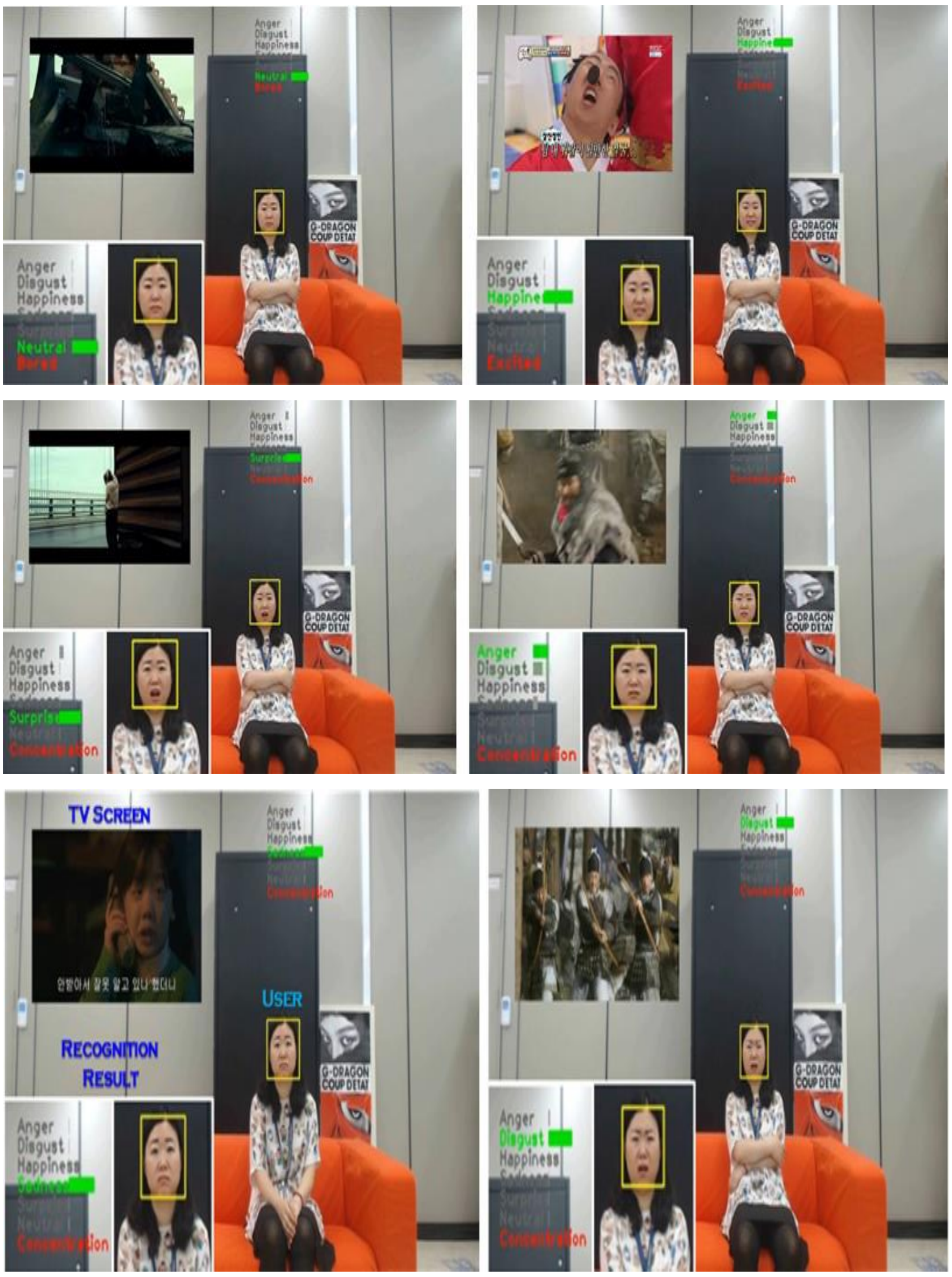


Figure 2.4 Facial Expressions

In [7], he authors propose a novel facial detection scheme based on the integration of YCgCr based skin colour segmentation and improved AdaBoost algorithm. Also, morphological operators are applied to improve the detection performance. Proposed method achieves accuracy of approximately 97% and has considerably good performance on images having complex background and can detect faces of various sizes, postures and expressions, under uncontrolled lighting environments. There are four approaches for face detection named: knowledge-based method, feature-based method, template matching method and methods based on statistical modelling.

The approach proposed by Viola and Jones significantly improves the rate and performance of face detection process and it is used for real-time applications most of the times. But the algorithm only makes use of the grey level features of human face. The proposed algorithm uses both skin colour feature and grey level features for facial detection. First, we separate the desired face region by skin colour-based detection, and then give it as the input to the improved Adaboost cascaded classifier, so that it could separate the person's face much more accurate.

In [8], the authors try to solve the problem of low-quality teaching in the universities by a real-time processing of classroom video through face detection technology and a software system to provide a basis for judging the quality of teaching. There are two important algorithms one is You Only Look Once (YOLO) algorithm is used to detect the number of classrooms. YOLO is a method of rapid object detection in deep learning, in terms of speed are far better than faster-RCNN, fast-RCNN and other networks.

The second one is the AdaBoost algorithm, an iterative algorithm for classifier, whose mainly purpose is to train several different classifiers which is to be called weak classifiers in the same training dataset. Then we join all the pre-training week classifiers together to form a stronger final classifier. This process can improve the model classification ability by continuous training.

In [9], the system proposed by the authors was tested on 5 employees who took several different positions while being photographed. A Python script was written using OpenCV interface to automatically generate new augmented images, as shown in Figure 2.5, out of the original ones. Next Dlib machine learning toolkit was used for marking the location of a person's nose, eyes, chin and mouth on the image.

Knowing the actual positions of these parts of a face on the image, a Python script automatically added random accessories: mustaches, glasses, etc. and creates new images for training dataset.

This led to the enlargement of the initial dataset and the improvement of the overall accuracy to 95.02%. Linear Support Vector Machine (SVM) was applied for this classification task. It overcomes the potential security issue of using Radio Frequency Identification (RFID) cards and Global Positioning System (GPS) as employees could forget the RFID card or the location device, or someone else could check in instead.

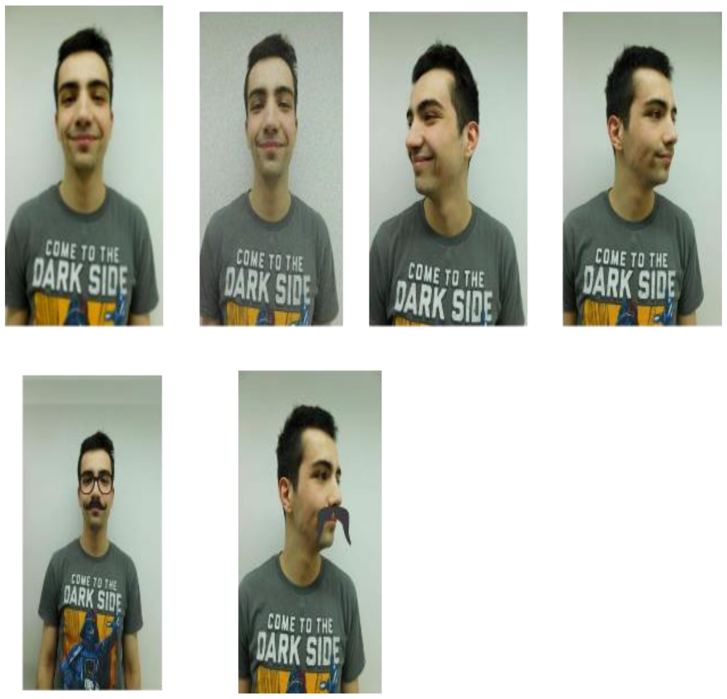


Figure 2.5 Examples of generated images

In [10], the authors present a configurable interactive embedding system to extract the facial image features in crowds. The proposed system generates unique facial embedding per person using the privacy-aware ResFace neural network and vectorized-l2-loss function in the training phase. Edge computing is then utilized to facilitate or expedite real-time decision making. The MKDE matching algorithm is applied to facial embedding pipelines generated by cameras at the edge for identity reidentification as well as security verification tasks.

In [11], the authors present a real-time face detection system using a moving camera. The proposed system consists of three modules, including Detection of face candidates: Face candidates are generated using the information of skin colour, edges, and face area. Verification of face candidates: HOG (Histogram of Oriented Gradient) features are generated from face candidates and a two-class C-SVM classifier with pretrained face samples is employed to determine whether face candidates are real faces or no. Face tracking: Overlapping area of two face targets in current and previous frames is estimated to determine whether the tracking will be continuous or not. . The data associated with a leaf cell varies by application, but the leaf cell represents a unit of interesting spatial information.

In [12], the authors propose a Real-time single-shot detector for face detection. It utilizes a single end-to-end deep neural network. It creates boundary boxes and utilizes those boxes to classify objects. Multi-scale feature mapping (mapping of where a certain kind of feature is found in the image; a high activation means a certain feature was found) overcome the difficulties of detecting small face.

In [13], the authors try to address the challenges in unconstrained real scene face detection during the night, we propose a real-time night face detector. First, it uses histogram equalization methods to process the input image of night to increase its contrast. Second, the image features called Promotion Normalized Pixel Difference (PRO-NPD) is proposed. PRO-NPD features are computed as the ratio of difference to sum between two-pixel values. Third, use a deep quadratic tree to learn the optimal subset of PRO-NPD features and their combinations.

A quadtree is a tree data structure in which each internal node has exactly four children. Quadtrees are the two-dimensional analog of octrees and are most often used to partition a two-dimensional space by recursively subdividing it into four quadrants or regions. The data associated with a leaf cell varies by application, but the leaf cell represents a unit of interesting spatial information.

In [14], the authors use a face detection framework based on cascaded convolutional neural network, which is used to balance the accuracy and running time cost. We use Full Convolutional Neural Network (FCNN) to extract candidate regions of human face in the first stage, which is more efficient than selective search, Edge Box and other algorithms. Combining with the Nom-Maximum Suppression (NMS) algorithm and bounding box regression during the whole process, we can get more accurate face position.

Non-max suppression is a way to eliminate points that do not lie in important edges. The Bounding Box Repressors' are essential because the initial region proposals might not fully coincide with the region that is indicated by the learned features of the Convolutional Neural Network. It is a kind of a refinement step. In this paper author explains about, a face detection approach based on cascaded convolution neural network is designed.

The approach achieves better performance by cascading three different convolution neural networks with high accuracy. Because the convolution neural network has a certain robustness to the faces of multi-gestures, occlusion, and illumination. Therefore, the approach designed in this paper has strong adaptability in face detection in complex situations. This paper has realized the mining method of the negative sample, which improves the accuracy of the model.

In [15], the authors propose a system that comprises of two modules defined at the hardware level which includes a Raspberry Pi Microcontroller with a few sensors connected to it and an Arduino Microcontroller with Global System for Mobile Communications (GSM) and Global Positioning System (GPS) capabilities installed together at the area of deployment. These two modules communicate with each other on the local network and together communicate with the users on the remote public network. This paper also focuses on how computer vision can be used to detect human activity at the site, use of deep convolutional neural networks to identify and match an image with a set of people authorized to visit a site, as shown in Figure 2.6 and Figure 2.7.

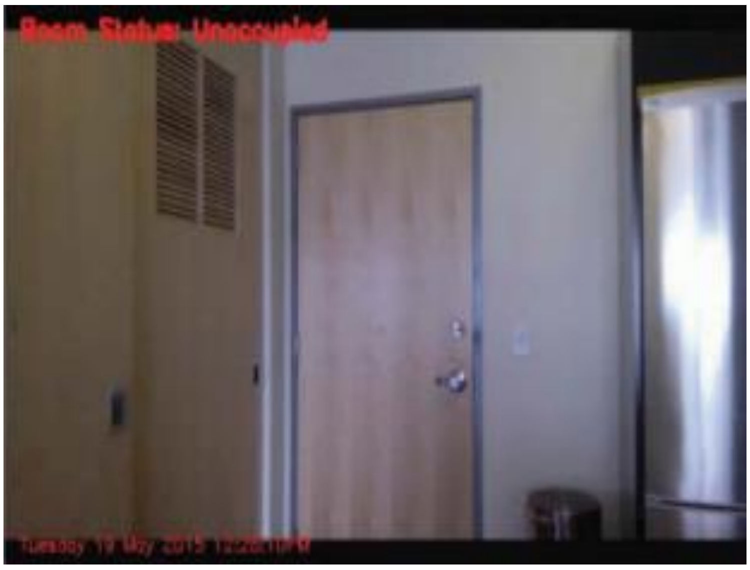


Figure 2.6 Frame from video stream showing Room is Unoccupied

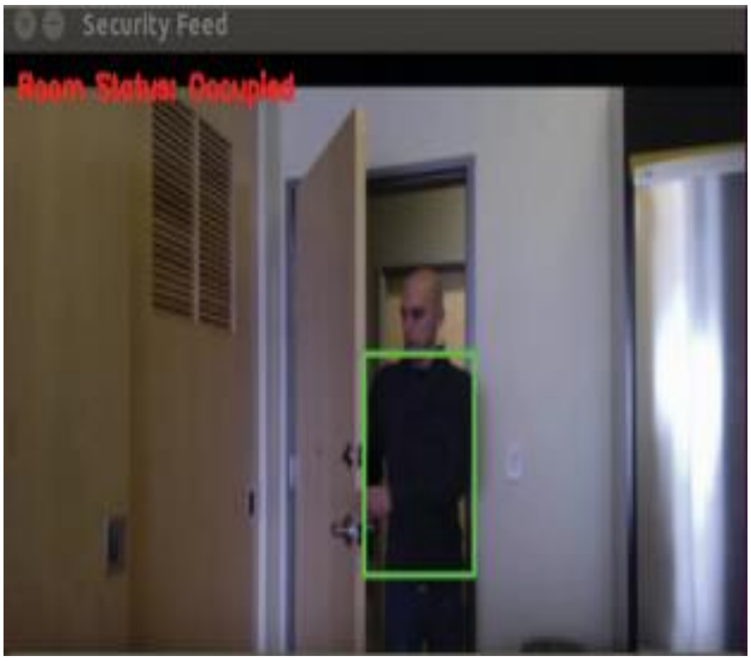


Figure 2.7 Frame from video stream showing Room is Occupied

CHAPTER 3

SYSTEM ANALYSIS

Digital image processing allows the use of much more complex algorithms, and hence, can offer both more sophisticated performance at simple tasks, and the implementation of methods which would be impossible by analog means. Digital Image Processing is the practical technology for:

- Classification
- Feature extraction
- Multi-scale signal analysis
- Pattern recognition
- Projection

3.1 Problem Identification

A problem can be regarded as a difference between the actual situation and the desired situation. This means that in order to identify a problem the team must know where it is meant to be and have a clear understanding of where it currently is in relation to the perceived problem.

3.1.1 Text Recognition from Image

Optical Character Recognition (also Optical Character Reader, OCR) is the electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast) . It is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining .

Some major challenges recognized are the font characteristics of the characters in paper documents and quality of images are some of the recent challenges. For good quality and high accuracy character recognition, OCR techniques expect high quality or high-resolution images. The surrounding scene makes it hard to segregate text from non-text, especially in uneven lighting and shadows. Uneven focus and blurring are observed when camera is in motion. There is very slight difference between some digits and letters for computers to recognize them and distinguish one from the others correctly. For example, it may not be very easy for computers to differentiate between digit “0” and letter “o”, especially when these characters are embedded in a very dark and noisy background. These challenges will be overcome by increasing the quality of frames from video using the approach of censoring incoming frames based on the Function of Rival Similarity (FRiS) function

3.1.2 Facial Recognition

Facial Recognition is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. They work by comparing selected facial features from given image with faces within a dataset. It is also described as a Biometric Artificial Intelligence based application that can uniquely identify a person by analyzing patterns based on the person's facial textures and shape. It is typically used as access control in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. It is widely adopted due to its contactless and non-invasive process. It has also become popular as a commercial identification and marketing tool. Other applications include advanced human-computer interaction, video surveillance, automatic indexing of images, and video dataset, among others.

A problem faced is that an efficient face recognition system must be able to deal with variations of face images in position illumination and expressions. Variations due to illumination and posture are always greater than those due to face identity. This makes real-time face recognition very challenging. To overcome this, the system will be built using a combination of HOG feature descriptor and SVM feature classifier, along with the Viola-Jones face detection framework, which is robust and has an efficiency, practical for real-time applications.

3.1.3 Video Content Analysis

Video content analysis (also video content analytics, VCA) is the capability of automatically analyzing video to detect and determine temporal and spatial events. This technical capability is used in a wide range of domains including entertainment, health-care, retail, automotive, transport, home automation, flame and smoke detection, safety and security. The algorithms can be implemented as software on general purpose machines, or as hardware in specialized video processing units.

Some of the main issues that limit the real-time application of video content analysis are limited flexibility of current content analysis software, hardware limitations for real-time processing of video signal and cost effectiveness of solutions. The Raspberry Pi kit provides a cost-effective solution and overcomes prior hardware limitations associated with real-time processing of video content. It is equipped with a camera, which will be interfaced with the proposed system, and, used to record as well as process livestreamed video.

3.2 Objectives

There is a potential security issue of using Radio Frequency Identification (RFID) cards and Global Positioning System (GPS) in organizations as employees could forget the RFID card or the location device, or someone else could check in to rooms instead.

The first objective is to overcome the problems associated with text recognition from image by enhancing the font characteristics of the characters and the quality of images. This will be achieved by increasing the quality of frames from video using the approach of censoring incoming frames based on the FRiS function. This will improve the ease in recognizing location tags.

The second objective is to overcome the difficulties in real-time face recognition by dealing with variations of face images in position, illumination and expressions. This will be achieved by building the system using robust and efficient methods like the combination of HOG feature descriptor and SVM feature classifier, and the Viola-Jones face detection framework. This will improve recognition of faces with various facial expressions and exposures to lighting.

The third objective is to overcome the limitations of hardware in real-time processing of video signals and cost effectiveness of solutions, that limit the effciiency of video content analysis. This will be achieved by using a cost-effective Raspberry Pi kit that comes equipped with a camera and can overcome the prior hardware limitations associated with real-time processing of video content.

3.3 Methodology

Methodology is the systematic, theoretical analysis of the methods applied to a field of study. It comprises the theoretical analysis of the body of methods and principles associated with a branch of knowledge. Typically, it encompasses concepts such as paradigm, theoretical model, phases and quantitative or qualitative techniques. A methodology does not set out to provide solutions—it is therefore, not the same as a method. Instead, a methodology offers the theoretical underpinning for understanding which method, set of methods, or best practices can be applied to a specific case, for example, to calculate a specific result. The methodology is the general research strategy that outlines the way in which research is to be undertaken and, among other things, identifies the methods to be used in it. These methods, described in the methodology, define the means or modes of data collection or, sometimes, how a specific result is to be calculated. Methodology does not define specific methods, even though much attention is given to the nature and kinds of processes to be followed in a particular procedure or to attain an objective.

3.3.1 System Overview

PHP, along with other Web technologies, like HTML5, CSS3 and JavaScript, will be used to create the Web based UI. The user will be able enter the name of the person to be found. In the camera's live stream, each person recognized in the room, by the Deep Learning application, will be stored in a dataset. The user will be provided with an additional option of viewing the names of all the other people that were sitting in the same room, at that same instant the person in question was found.

The Apache Web server will be used to provide the End-user form and interact with the backend application. At the backend, a Python IDE will be used to run the Deep Learning application program. The Keras API, using the Tensorflow library as a backend, will be used to design the Deep Neural Network (DNN) that will be trained in person recognition. OpenCV and libraries will be used to process the frames of video, extracted from the camera's live stream .

3.3.2 Front-end Design

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.

The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It does not include peripherals (such as keyboards and mice) and cases. However, some accessories have been included in several official and unofficial bundles. The Raspberry Pi kit will be equipped with a camera, which will be interfaced with the proposed system and used to record livestreamed video.

HTML5, CSS3 and JavaScript, along with PHP, will be used to create the front-end UI. HTML5 is a markup language used for structuring and presenting content on the World Wide Web. Its goals are to improve support for the latest multimedia and other new features.

Cascading Style Sheets version 3 (CSS3) is a style sheet language used for describing the presentation of a document written in a markup language like HTML5. CSS3 is designed to enable the separation of presentation and content, including layout, colors, and fonts. Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices.

PHP, which stands for Hypertext Preprocessor (or simply PHP) is a server-side scripting language designed for Web development, and also used as a general-purpose programming language. PHP code will be embedded into HTML5 code. PHP code will be processed by a PHP interpreter implemented as a module in the Apache Web. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page.

3.3.3 Back-end Design

WampServer refers to a software stack for the Microsoft Windows operating system, consisting of the Apache web server, OpenSSL for SSL support, MySQL database and PHP programming language. MySQL is an open-source relational database management system (RDBMS). MySQL is a central component of the WAMP open-source web application software stack. MySQL is written in C and C++. It has also been tested to be a fast, stable and true multi-user, multi-threaded SQL database server.

The dataset of images required for training and testing the system will be acquired manually and stored in a directory on the web server. As the Deep Learning application runs and video is livestreamed through the Raspberry Pi kit's camera, the location tags, as well as names of those whose faces are recognized (new faces will be stored in the dataset as 'New Face 1', etc), will be added to the dataset.

Based on the user's input for the name of the person to be located, the appropriate location tag entry will be retrieved. Also, the names of the other occupants seated in the room corresponding to the recently returned location tag will be displayed.

Face recognition quality from video will be increased using the approach of censoring incoming photos based on Function of Rival Similarity (FRiS) function. The system will be using a combination of Histogram of Oriented Gradients (HOG) feature descriptor and SVM feature classifier along with the Viola-Jones face detection framework. Jones object detection framework will be used for face detection and HOG feature descriptor will be used for face feature extraction. Support Vector Machines (SVMs) which are powerful tools, will be used for developing pattern classification and function approximation systems for face identification.

The system will use an integration of YCgCr based skin colour segmentation and improved AdaBoost algorithm. Morphological operators will also be applied to improve the detection performance. It will use both skin colour feature and grey level features for facial detection.

The desired face region will be separated by skin colour-based detection, and then will be given as the input to the improved Adaboost cascaded classifier, so that it can separate the person’s face much more accurately. The proposed method will achieve an accuracy of approximately 97% and will have a considerably good performance on images having complex background and can detect faces of various sizes, postures and expressions, under uncontrolled lighting environments.

Eigen face algorithm will be used in face recognition to make this system faster and efficient. When image of a face is taken, it will get centered by using the eyes. This algorithm will follow a process through which the face is divided into smaller parts called as Eigen vectors, that are used in a process of calculation.

A Python script will be written using OpenCV interface to automatically generate new augmented images out of the original ones. Next, the Dlib machine learning toolkit will be used for marking the location of a person’s nose, eyes, chin and mouth on the image.

Knowing the actual positions of these parts of a face on the image, a Python script will automatically add random accessories like mustaches, glasses, etc, and create new images for training dataset. This will lead to the enlargement of the initial dataset and the improvement of the overall accuracy.

A face detection approach based on Convolution Neural Network will be designed . The approach will achieve a better performance by cascading three different Convolution Neural Networks with high accuracy. The Convolution Neural Network will have a certain robustness to the faces of multi-gestures, occlusion, and illumination.

3.4 System Requirements Specification

A System Requirements Specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function. Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure.

3.4.1 Hardware Requirements Specification

- 64-bit CPU
- 4-8 GB Memory
- Built-in camera

3.4.2 Software Requirements Specification

- OS (eg. Windows 10)
- PHP
- CSS3
- HTML5
- Apache Web server
- JavaScript

3.4.3 Functional Requirements Specification

A Functional Requirement defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs.

3.4.3.1 Functional requirements - User

Users should be able to view the room(s) a person is located in and all other occupants of the room(s), whether they have known or unknown faces. The timestamps for each occupant's location should also be provided.

3.4.4 Non-Functional Requirements Specification

A Non-Functional Requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually architecturally significant requirements.

3.4.4.1 Security requirements

It is of utmost importance to ensure that there is protection against unauthorized access to the blog posts. Users must be provided with a login ID and password which grant access only to their respective account. Similarly, every user can only view their own posts and posts of those whom they follow by them. The Admin on the other hand has privileged access in order to view all data and ensure seem less and efficient experience to all of its users.

3.4.4.2 Performance requirements

The PCs used must be at least be INTEL CORE i7 machines so that they can give optimum performance of the product. In addition to these requirements, the system should also embrace the following requirements:-

- Reliability: The system should have little or no downtime.
- Ease of Use: The general and administrative views should be easy to use and intuitive.

3.4.4.3 Design and Interface requirements

The designers must design the dataset is such a way that any updates should be saved effectively in the dataset. The interface which is provided in this software allows users to view the names of the other occupants that were located in the same room(s), at the same time the face of the person in question was recognized. The front-end is connected with a socket connection, established through PHP, which has inbuilt functions for socket connectivity.

CHAPTER 4

SYSTEM DESIGN

System Design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

4.1 System Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).

The system consists of the user and the Person Location application. In turn, the application is made up of two modules. The Face Detection and Recognition Module is responsible for implementing the logic for detection of faces in a video stream and their real-time recognition using the CNN pre-trained on the dataset of face images. The Text Extraction and Recognition Module is responsible for implementing the logic for detection of text in still frames of a video stream and their real-time recognition using the CNN pre-trained on the dataset of text images.

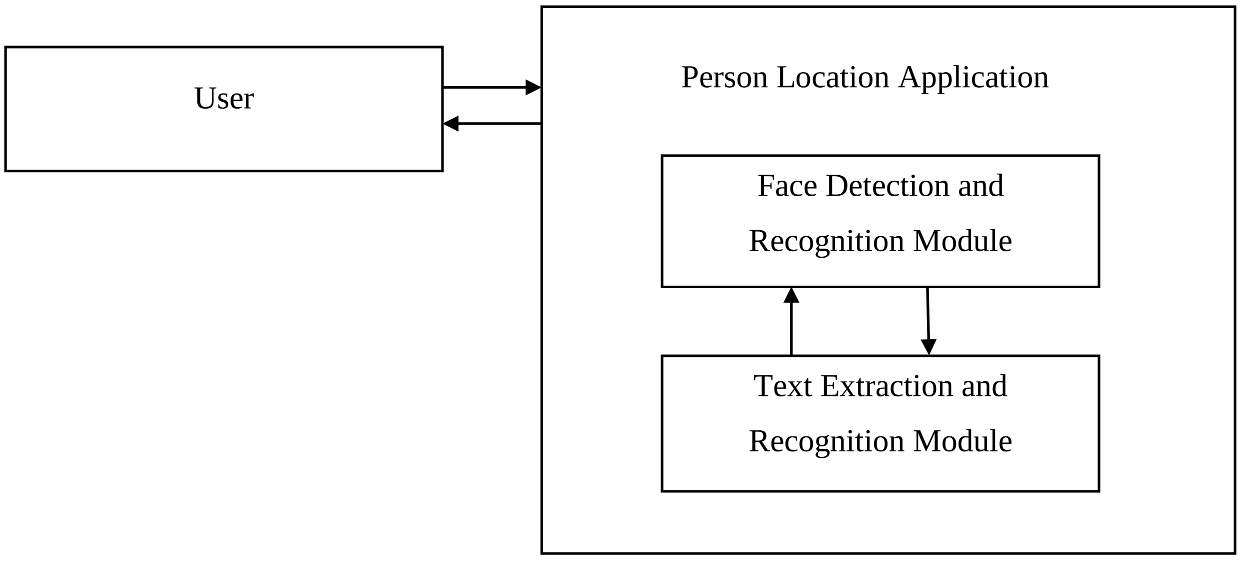

User
Person Location Application
Face Detection and Recognition Module
Text Extraction and Recognition Module

Figure 4.1 System Architecture

4.2 Detailed Design

Detailed design is a stage in the computer system design, specification and development process. It refers to the development stage during which the actual implementation design (the how to do what) is determined and documented. The process is highly iterative - parts of the process often need to be repeated many times before another can be entered - though the part(s) that get iterated and the number of such cycles in any given project may vary.

4.2.1 High Level Design

High-level design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces.

The HLD uses possibly nontechnical to mildly technical terms that should be understandable to the administrators of the system. In contrast, low-level design further exposes the logical detailed design of each of these elements for programmers. The application is made up of two modules.

- The Face Detection and Recognition Module
- The Text Extraction and Recognition Module

4.2.2.1 Face Detection and Recognition Module

The module uses the dataset of stored facial images to perform facial recognition, as shown in Figure 4.2. To recognize faces that are not part of any current, we gather examples of faces we want to recognize and then quantify them in some manner. This process is typically referred to as facial recognition enrollment.

Before the CNN is trained, the images from the dataset will be encoded in batches (based on the sub-directory labels), into 128-d vectors. Then, an XML classifier is loaded. The classifier is used by the CNN for training on stored face encodings.

Then, when the input frames from the video are loaded in grayscale mode, the pre-trained CNN will perform matching with the new face encodings for prediction of the label for the faces. The input frame is modified with a bounding rectangular box around the detected face, with the predicted label written on it, to the top-left corner, as shown in Figure 4.3.

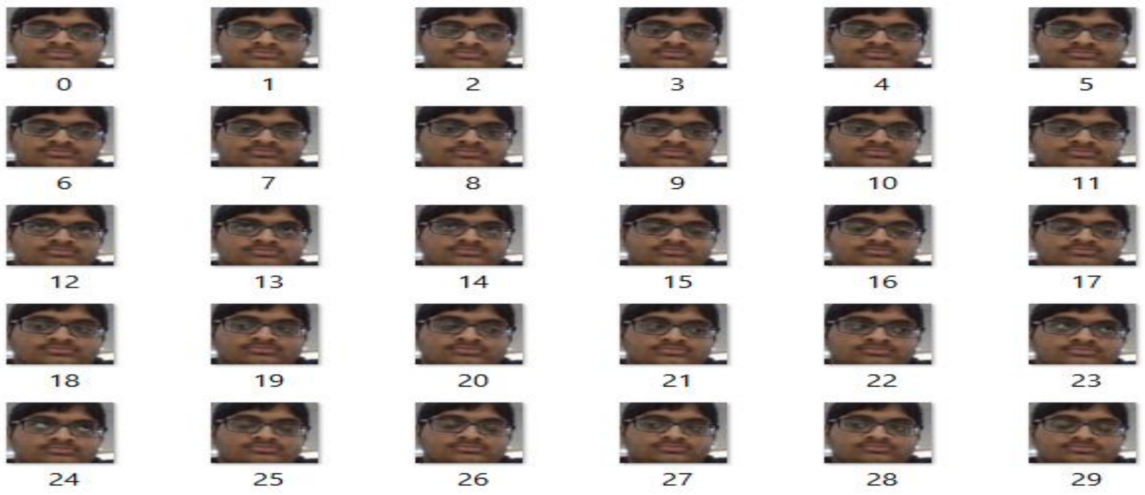


Figure 4.2 Faces dataset

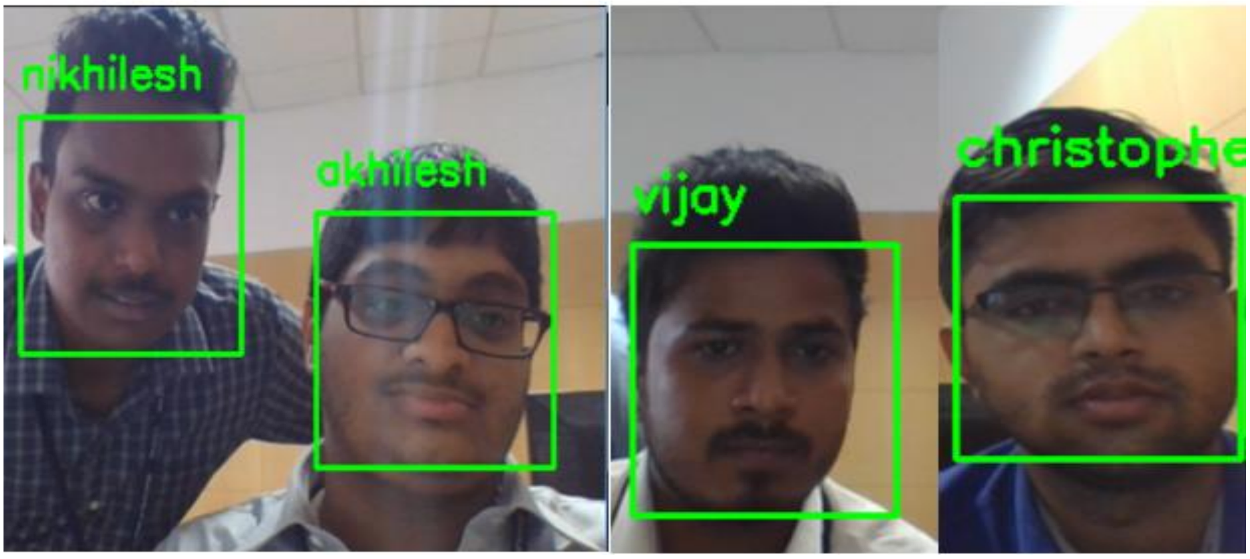


Figure 4.3 Output of Face Detection and Recognition Module

4.2.2.2 Text Extraction and Recognition Module

The module uses the camera to receive the input video stream. It then extracts each frame from the video in order to detect whether there is text in it or not. Then, the pre-trained CNN model is loaded. When the contours in the input frames from the video are converted to grayscale, the pre-trained CNN will perform matching on each contour, one by one, and predict the label for each of them.

The prediction results for each contour is combined to produce the final prediction result for the room sign(s). The input frame is modified with a bounding rectangular box around each detected text, and the predicted labels are written on them, to their top-left corners, as shown in Figure 4.4.

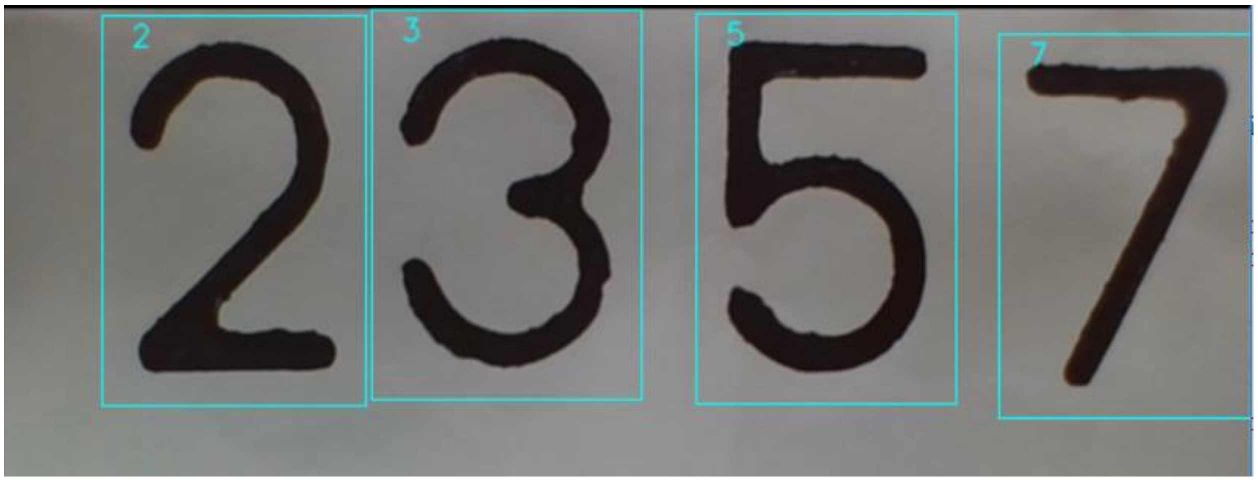


Figure 4.4 Output of Text Extraction and Recognition Module

4.2.2 Low Level Design

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work. Post-build, each component is specified in detail. The LLD phase is the stage where the actual software components are designed. During the detailed phase the logical and functional design is done and the design of application structure is developed during the high-level design phase.

4.2.1.1 Face Detection and Recognition Module

The process of creating a custom dataset of example faces uses a webcam to detect faces in a video stream and save the example face images/frames to disk. Using OpenCV and a webcam it’s possible to detect faces in a video stream and save the examples to disk. This process can be used to create a face recognition dataset.

To gather example face images of people, we detect the (x, y)-coordinates of their face in a video stream and write the frames containing their face to disk. This process is performed over multiple days or weeks, to create a more diverse set of images representative of that particular person’s face.

Examples of each face is gathered with:

- Different lighting conditions
- Times of day
- Moods and emotional states

The dataset contains sub-directories for each person the facial recognition system can recognize. Before the CNN is trained, the images from the dataset will be encoded in batches (based on the sub-directory labels), into 128-d vectors. These face encodings (128-d vectors, one for each face) are stored in a .pickle file. OpenCV comes with a trainer as well as detector. It already contains a pre-trained classifier for faces. Here, the haarcascade_frontalface_default.xml file is used for detection and localization of faces in frames.

As shown in Figure 4.5, the module uses the camera to receive the input video stream. It then extracts each frame from the video in order to detect whether there are faces in it or not. When faces are detected in frames of the video, they are converted into 128 dimensions facial embeddings, like the ones on which the CNN is trained.

The computed embeddings are fed as input to the pre-trained CNN, which combines it with stored facial embeddings in order to recognize and predict a match. A tolerance factor is set considering external environment factors, like illumination. This sets an upper bound on the amount of deviation allowed between the new and stored facial embeddings.

Though the CNN predicts probabilities for each possible match, it chooses the matched name with highest probability as its prediction.

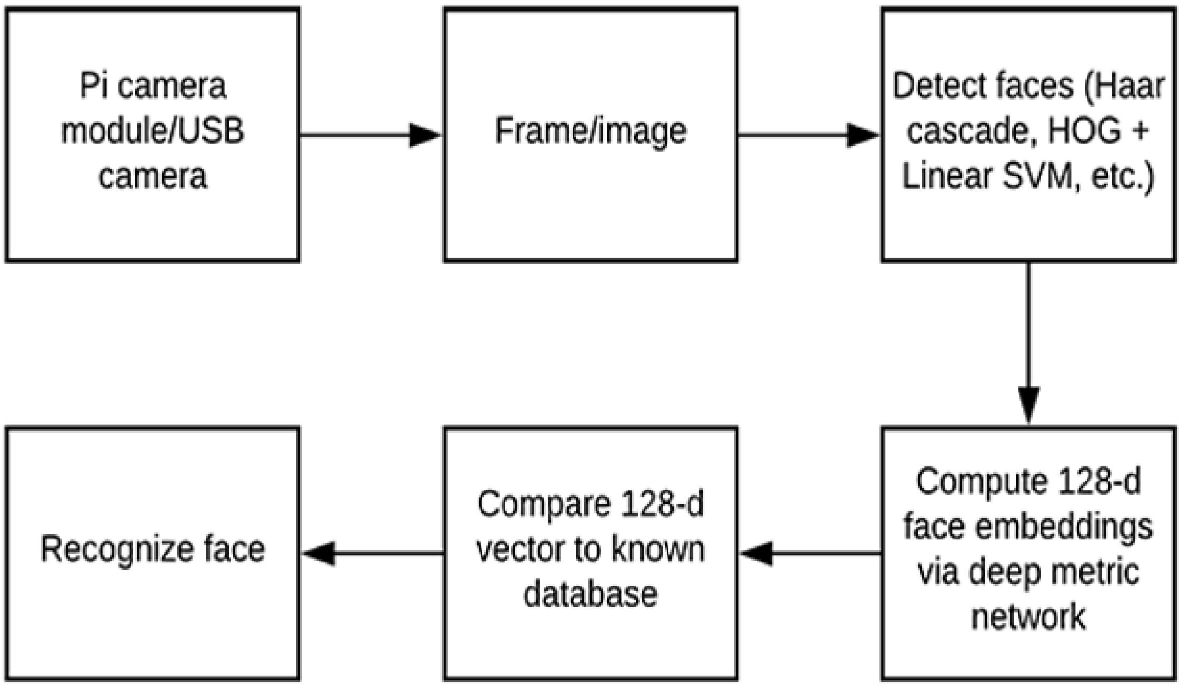

Pi camera module/USB camera
Frame/image
Detect faces (Haar cascade, HOG + Linear SVM, etc.)
Compute 128-d face embeddings via deep metric network
Compare 128-d vector to known database
Recognize face

Figure 4.5 Face Detection and Recognition Module

4.2.1.2 Text Extraction and Recognition Module

The module uses the dataset of stored text images to perform text recognition. The dataset contains sub-directories for each type of text the text recognition system can recognize. A CNN is trained on the images from the dataset, and the trained model along with its learned weights are saved to disk.

The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by "re-mixing" the samples from NIST's original datasets. The black and white images from NIST were normalized to fit into a 28x28 pixel bounding box and anti-aliased, which introduced grayscale levels, as shown in Figure 4.6.

The MNIST database contains 60,000 training images and 10,000 testing images. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset. A hierarchical system of convolutional neural networks, manages to get an error rate on the MNIST database of 0.23%. The original creators of the database keep a list of some of the methods tested on it. An extended dataset similar to MNIST called EMNIST has been published in 2017, which contains 240,000 training images, and 40,000 testing images of handwritten digits and characters.

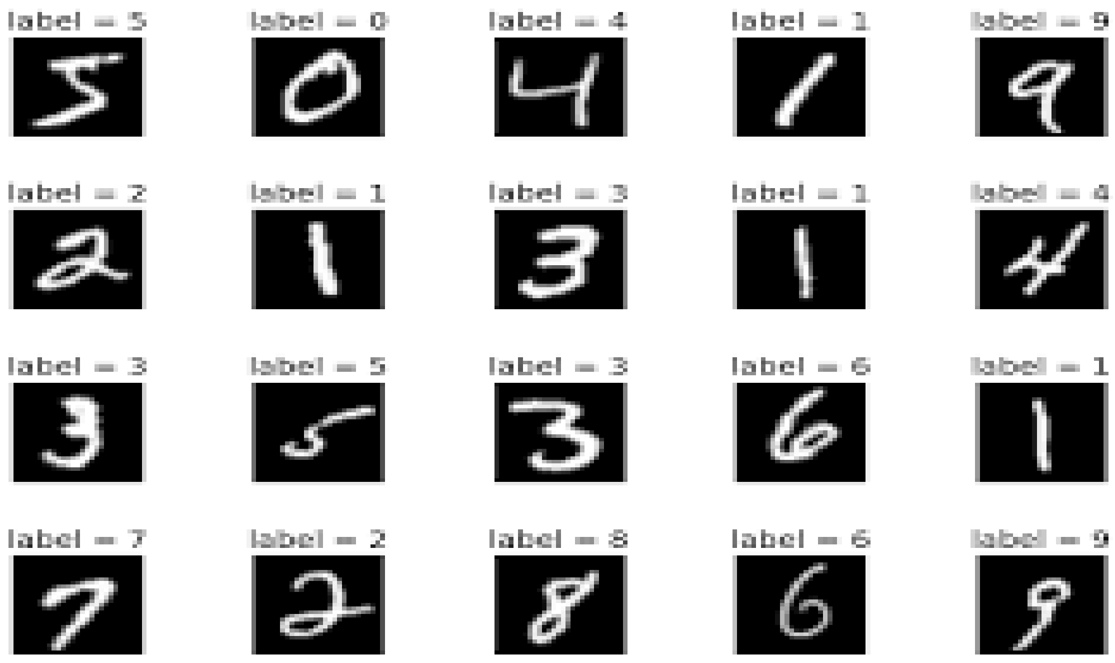


Figure 4.6 Input to Text Extraction and Recognition Module

As shown in Figure 4.7, when contours of text are detected in frames of the video, using the adaptive thresholding algorithm, they are converted to grayscale, like the images on which the CNN is trained. The modified contours are fed as input, one by one, to the pre-trained CNN, which uses its training to predict a match. Though the CNN predicts probabilities for each possible match, it chooses the matched text with highest probability as its prediction. Once all the contours are recognized, the results of the predictions for each contour is combined and returned as the final prediction result.

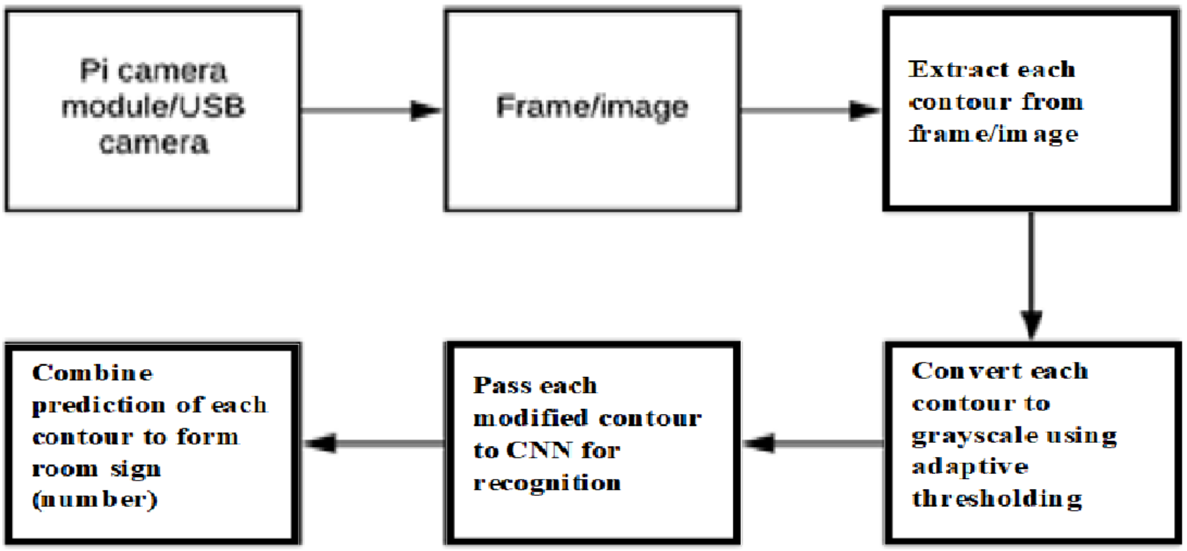

Pi camera module/USB camera
Frame/image
Extract each contour from frame/image
Convert each contour to grayscale using adaptive thresholding
Pass each modified contour to CNN for recognition
Combine prediction of each contour to form room sign (number)

Figure 4.7 Text Extraction and Recognition Module

4.3 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). It 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. The system provides an interface between the user and the backend business logic. The user can provide the system with the name of the person who is to be located and whose face is to be recognized.

In addition, the user is provided with the option to view the names of other occupants of the room(s), at the same instant when the face of the person in question was detected. The backend consists of a trained deep neural network, training and testing datasets used, and a file storing the names of the people whose faces have been recognized so far. Figure 4.8 illustrates the flow of data in this Person Location system.

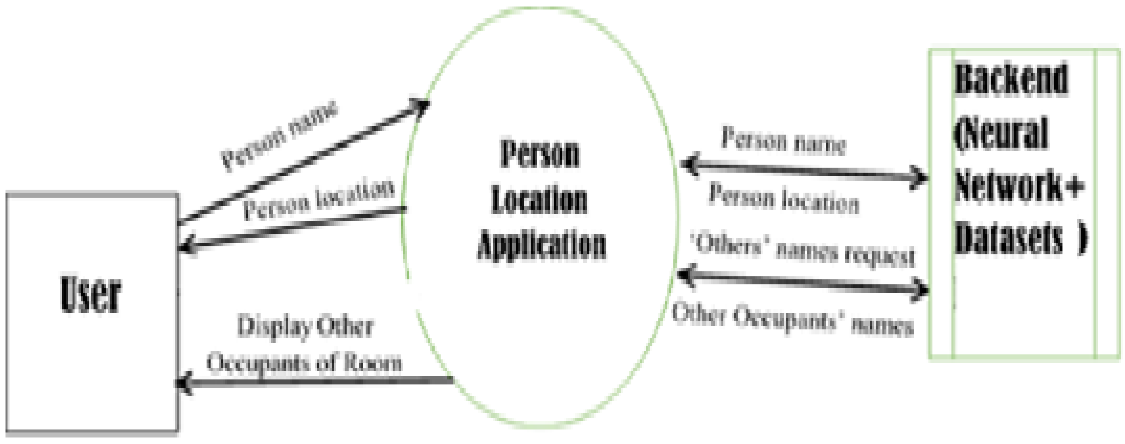


Figure 4.8 System Data Flow Diagram

4.4 Use Case Diagram

A use case diagram (UCD) at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

A use-case diagram can help provide a higher-level view of the system. They are the blueprints for your system. They provide the simplified and graphical representation of what the system must actually do. They provide a view for the stakeholder to understand how the system is going to be designed. A use case diagrams conveyS the intent of the system in a more simplified manner to stakeholders. The purpose of the use case diagrams is simply to provide the high level view of the system and convey the requirements in laypeople's terms for the stakeholders.

The user can interact with the system. It involves the user providing the system with the name of the person whose face is to be recognized. The user can then also view the names of the others that were located in the same room(s), at the same time the face of the person in question was recognized. Figure 4.9 illustrates these available user functionalities.

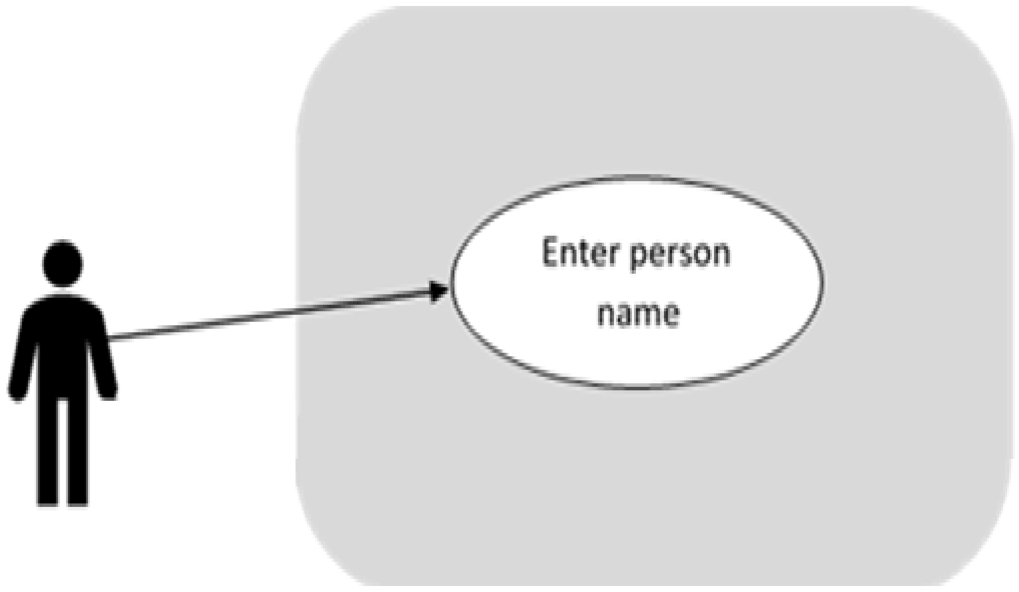


Figure 4.9 System use case diagram

4.5 Sequence Diagram

A sequence diagram as shown in Figure 4.10 shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

Figure 4.10 illustrates the sequence of interactions between the user and the system. The user first provides the system with the name of the person whose face is to be recognized. The application begins by retrieving some resources required for it to run a script. This includes the dataset of stored face images needed for face recognition, as well as, a trained CNN to perform text recognition.

Once the script finishes execution and outputs the results, the user is returned the location of the person whose name was entered. The user is also provided with the names of the other occupants that were located in the same room(s), at the same time the face of the person in question was recognized.

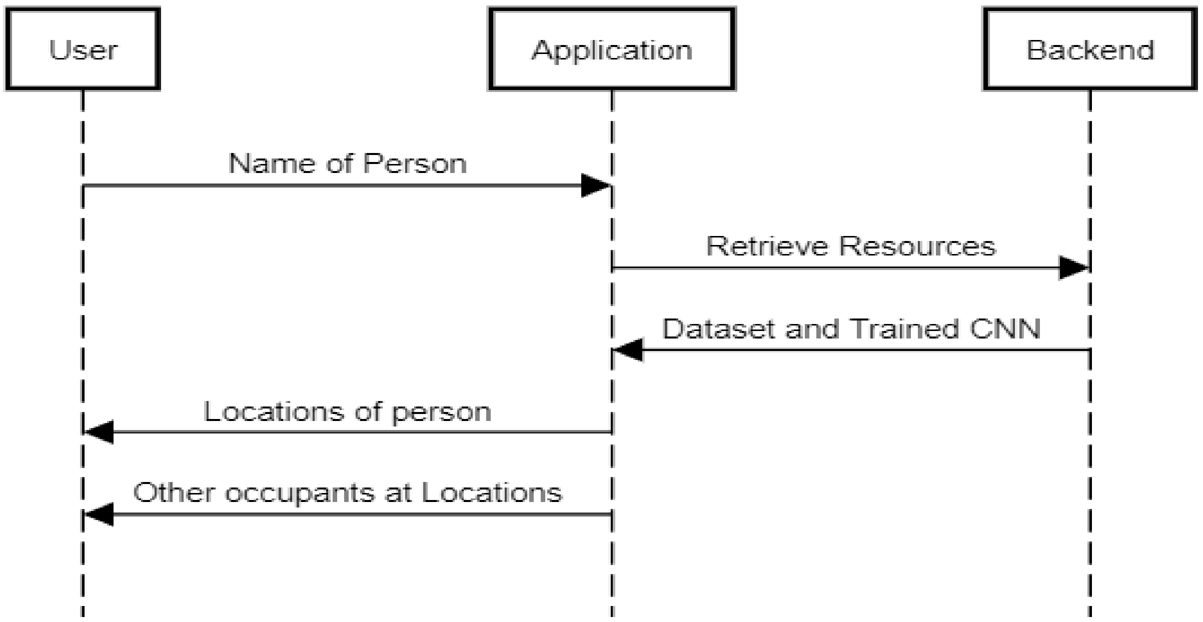


Figure 4.10 System sequence diagram

CHAPTER 5

IMPLEMENTATION

Implementation is the process of defining how the system should be built, ensuring that it is operational and meets quality standards. It is a systematic and structured approach for effectively integrating a software-based service or component into the requirements of end users.

5.1 Overview of System Implementation

Systems implementation is the process of: defining how the information system should be built (i.e., physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e., quality assurance).

5.1.1 Front-end – HTML5 and CSS3

HTML5 is a software solution stack that defines the properties and behaviors of web page content by implementing a markup based pattern to it.

Cascading Style Sheets (CSS) 3 is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.

This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

5.1.2 Back-end – Python and PHP (WAMP server)

WampServer refers to a software stack for the Microsoft Windows operating system, created by Romain Bourdon and consisting of the Apache web server, OpenSSL for SSL support, MySQL database and PHP programming language. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable.

The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

Python is an interpreted, high-level, general-purpose programming language. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural. It also has a comprehensive standard library. Python interpreters are available for many operating systems.

5.2 Algorithms

An algorithm is a step by step method of solving a problem. It is commonly used for data processing, calculation and other related computer and mathematical operations. An algorithm is also used to manipulate data in various ways, such as inserting a new data item, searching for a particular item or sorting an item. An algorithm is a detailed series of instructions for carrying out an operation or solving a problem.

In a non-technical approach, we use algorithms in everyday tasks, such as a recipe to bake a cake or a do-it-yourself handbook. Technically, computers use algorithms to list the detailed instructions for carrying out an operation.

For example, to compute an employee’s paycheck, the computer uses an algorithm. To accomplish this task, appropriate data must be entered into the system. In terms of efficiency, various algorithms are able to accomplish operations or problem solving easily and quickly.

5.2.1 Haar Cascade Classifier (Viola-Jones Algorithm)

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.

Consider working 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 Figure 5.1 are used.

They are just like the 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.

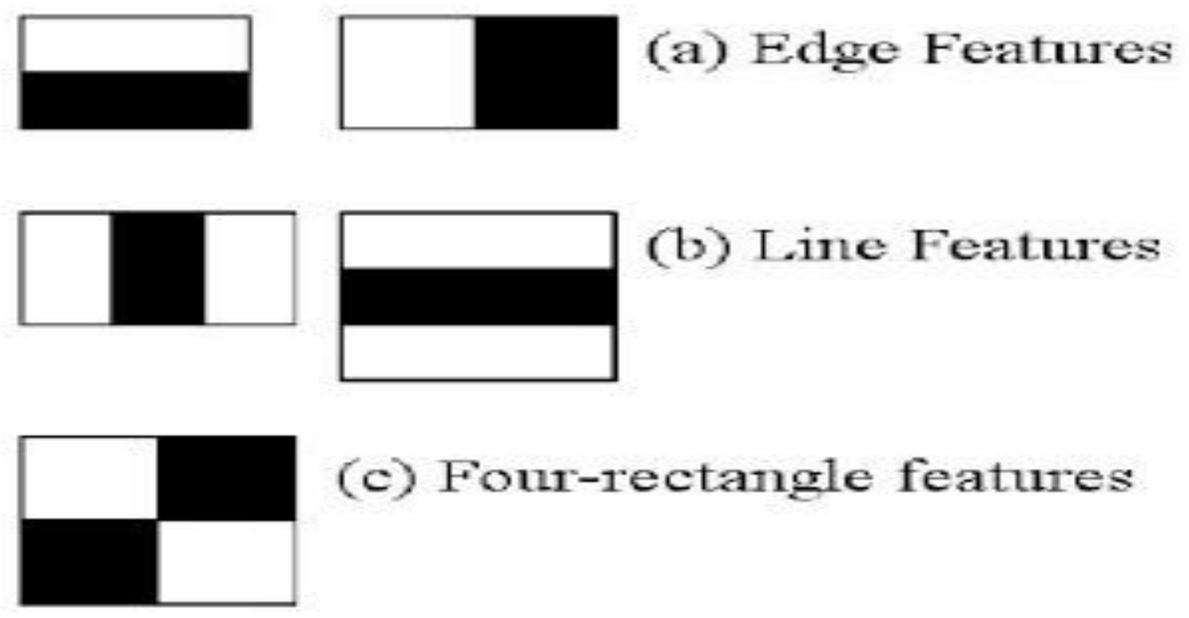


Figure 5.1 Haar features

All possible sizes and locations of each kernel are used to calculate lots of 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. It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the figure 5.2 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. We select the best features by Adaboost.

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. Historically, working with only image intensities (i.e., the RGB pixel values at each and every pixel of image) made the task of feature calculation computationally expensive. Viola and Jones[1] adapted the idea of using Haar wavelets and developed the so-called Haar-like features.

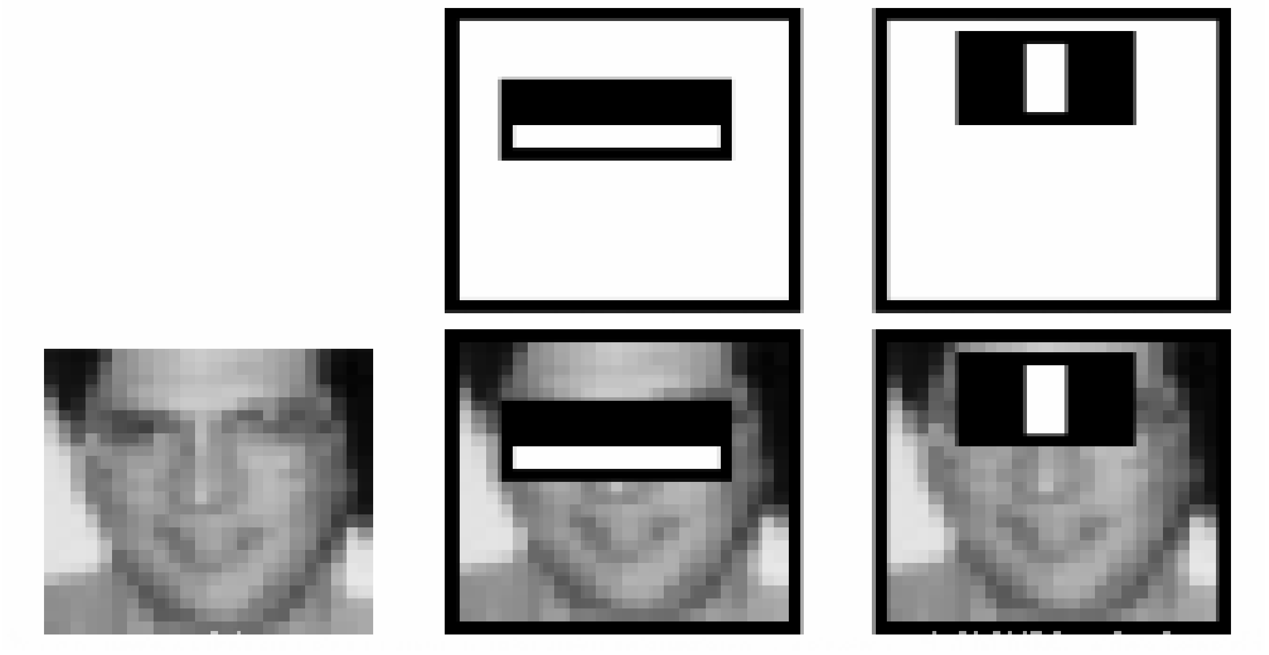


Figure 5.2 Good features

For this, each and every feature is applied 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. A reduction from 160000+ features to 6000 features is a big gain.

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. The authors' detector had 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in the first five stages. According to the authors, on average 10 features out of 6000+ are evaluated per sub-window.

5.2.2 Adaptive Thresholding

Thresholding is used to segment an image by setting all pixels whose intensity values are above a threshold to a foreground value and all the remaining pixels to a background value. Whereas the conventional thresholding operator uses a global threshold for all pixels, adaptive thresholding changes the threshold dynamically over the image. This more sophisticated version of thresholding can accommodate changing lighting conditions in the image, e.g. those

Adaptive thresholding typically takes a grayscale or color image as input and, in the simplest implementation, outputs a binary image representing the segmentation. For each pixel in the image, a threshold has to be calculated. If the pixel value is below the threshold it is set to the background value, otherwise it assumes the foreground value. There are two main approaches to finding the threshold: (i) the Chow and Kaneko approach and (ii) local thresholding. The assumption behind both methods is that smaller image regions are more likely to have approximately uniform illumination, thus being more suitable for thresholding.

Chow and Kaneko divide an image into an array of overlapping subimages and then find the optimum threshold for each subimage by investigating its histogram. The threshold for each single pixel is found by interpolating the results of the subimages. The drawback of this method is that it is computational expensive and, therefore, is not appropriate for real-time applications.

An alternative approach to finding the local threshold is to statistically examine the intensity values of the local neighborhood of each pixel. The statistic which is most appropriate depends largely on the input image. The size of the neighborhood has to be large enough to cover sufficient foreground and background pixels, otherwise a poor threshold is chosen.

On the other hand, choosing regions which are too large can violate the assumption of approximately uniform illumination. This method is less computationally intensive than the Chow and Kaneko approach and produces good results for some applications.

Like global thresholding, adaptive thresholding is used to separate desirable foreground image objects from the background based on the difference in pixel intensities of each region.

Global thresholding uses a fixed threshold for all pixels in the image and therefore works only if the intensity histogram of the input image contains neatly separated peaks corresponding to the desired subject(s) and background(s). Hence, it cannot deal with images containing, for example, a strong illumination gradient.

Local adaptive thresholding, on the other hand, selects an individual threshold for each pixel based on the range of intensity values in its local neighborhood. This allows for thresholding of an image whose global intensity histogram doesn't contain distinctive peaks. A task well suited to local adaptive thresholding is in segmenting text from the image.

5.3 Pseudocode

Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm. It uses the structural conventions of a normal programming language, but is intended for human reading rather than machine reading. Pseudocode typically omits details that are essential for machine understanding of the algorithm, such as variable declarations, system-specific code and some subroutines. The programming language is augmented with natural language description details, where convenient, or with compact mathematical notation. The purpose of using pseudocode is that it is easier for people to understand than conventional programming language code.

5.3.1 Gathering the faces dataset

Before face recognition can be applied, the dataset of people's images we want to recognize was gathered. This is done by performing face enrollment using a camera, along with face detection to gather the faces. The algorithm is rerun for each person's registration.

```
1. Load OpenCV's Haar cascade for face detection from disk
2. Initialize the video stream and allow the camera to warm up.
3. Loop over the frames from the video stream
   Until 30 images are captured
     • Grab the frame from the video stream
     • Detect the face in the grayscale frame
     • Write the frame to the corresponding folder in the dataset
```

5.3.2 Computing face recognition embedding

A deep, convolutional neural network is used to compute the 128-d vectors (lists of 128 floating point values) that quantify each face in the dataset.

```
1. Import the necessary packages
```

2. Grab the paths to the input images in the dataset
3. Initialize the list of known encodings and known names
4. Loop over the image paths

   For each path in the list of image paths

   - Extract the person name from the image path
   - Load the input image and convert it from BGR to RGB
   - Localize and detect the (x, y)-coordinates of the bounding boxes corresponding to each face in the input image
   - Compute the facial embedding for the face
5. Loop over the encodings

   For encoding in encodings

   - Add each encoding + name to the set of known names and encodings
6. Save the facial encodings + names to encodings.pickle file

5.3.3 Creating a trained text recognition model

1. Import the MNIST dataset
2. Preprocess the data
3. Build the model
   - Setup the layers
   - Compile the model
   - Train the model
4. Save the trained model and it's weights to a .mnist file on disk

5.3.4 Recognize room signs in frames of video streams

1. Import the necessary packages
2. Load the trained text recognition model from disk.
3. Initialize the video stream for text detection and allow the camera sensor to warm up
4. Loop over the contours in the frame from the video file stream

   For each contour

   - Use trained cnn and predict a label
   - Write the label on the frame
5. Show the modified frame with boxes around each detected piece of text.

5.2.5 Recognize faces in video streams

1. Import the necessary packages

2. Load the known faces and embeddings along with OpenCV's Haar cascade for face detection
3. Initialize the video stream for face detection and allow the camera sensor to warm up
4. Loop over frames from the video file stream

- Grab the frame from the threaded video stream and resize it to speedup processing
- Convert the input frame from (1) BGR to grayscale (for face detection) and (2) from BGR to RGB (for face recognition)
- Detect faces in the grayscale frame
    - If face is Unknown, click 30 snaps and create another subdirectory for it under dataset/
    - Compute facial encodings and create new encodings.pickle file
    - Go to step 4
- Compute the facial embeddings for each face bounding box
- Loop over the facial embeddings
- Attempt to match each face in the input image to the known encodings
- Find the indexes of all matched faces then initialize a dictionary to count the total number of times each face was matched
- Loop over the matched indexes and maintain a count for each recognized face
- Determine the recognized face with the largest number of votes
- Update the list of names
- Loop over the recognized faces
    - For each face in the list of faces
- Draw the predicted face name on the image

5.3.6 System pseudocode

1. Import the necessary packages
2. Load the known faces and embeddings along with OpenCV's Haar cascade for face detection and the trained cnn model for text recognition.
3. Initialize the video stream for face detection and allow the camera sensor to warm up
4. Loop over frames from the video file stream
    - Grab the frame from the threaded video stream and resize it to speedup processing
    - Convert the input frame from (1) BGR to grayscale (for face detection) and (2) from BGR to RGB (for face recognition)
    - Detect faces in the grayscale frame
    - Compute the facial embeddings for each face bounding box
    - Loop over the facial embeddings

- If face is Unknown,
    - Click 30 snaps and create another subdirectory for it under dataset/
    - Compute facial encodings and create new encodings.pickle file
    - Go to step 4
- Attempt to match each face in the input image to the known encodings
- Find the indexes of all matched faces then initialize a dictionary to count the total number of times each face was matched
- Loop over the matched indexes and maintain a count for each recognized face

- determine the recognized face with the largest number of votes
- Update the list of names
- Loop over the recognized faces
  For each face in the list of faces

- Draw the predicted face name on the image
- Display the image to the screen

- If key ‘p’ is pressed, start a video stream for text detection. When key ‘c’ is pressed, detect numbers in the room sign in the frame. When key ‘a’ is pressed, stop detection.
- Update the list of rooms
- Go to step 4 if the key ‘x’ is not pressed.
- If the key ‘x’ is pressed, write the name and room details to a text file, and save a frame for each detected face.

5.4 Implementation Support

Implementation support. While it is true that successful construction is based on sound planning, it also requires smooth implementation. Technical expertise is necessary for a quality implementation, while methodological competence guarantee that the project runs according to schedule.

5.4.1 Python Libraries and Functions

1. **imutils:** It provides a series of modules with convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much more easier with OpenCV.

    - **video.VideoStream:** It is used to access the camera of the device streaming live videos.
    - **path.list_images:** It is used to obtain the absolute paths of the location of face images in the dataset, stored in their respective sub-folders.

2. **keras:** It is a neural network library.

   - **models.load_model():** It is used to load the trained CNN and it's weights.
   - **models.predict_classes():** It is used to predict the text in an image.

3. **opencv-python:** It provides the cv2 module with many functions for computer vision.

   - **cvtColor():** It converts an image from one color space to another (color to grayscale).
   - **GaussianBlur():** The image is convolved with a Gaussian filter, a low-pass filter that removes the high-frequency components.
   - **adaptiveThreshold():** In thresholding, if a pixel value is greater than a threshold value, it is assigned one value (white), else it is assigned another value (black). This function calculates the threshold for small regions of the image, so we get different thresholds for different regions of the same image and it gives us better results for images with varying illumination.
   - **resize():** It resizes images for input to the trained CNN.
   - **findContours():** It finds independent contours in an image, which are curves joining all the continuous points (along the boundary), having same color or intensity. The contours are used for object detection and recognition.
   - **CascadeClassifier():** It is used to create a detector based on a classifier (here, Haar Cascade Classifier).
   - **detectMultiScale():** It is used to detect faces in the image using the detector.
   - **imread():** It is used to load an image.
   - **boundingRect():** It gives the coordinates of the corner points of the rectangle enclosing contours.
   - **putText():** It is used to put text in a video frame.
   - **rectangle():** It is used to draw a rectangle in a video frame.
   - **imshow():** It is used to draw a modified video frame.

- **waitKey():** It is used to wait for a key press.
- **destroyAllWindows():** It is used to destroy stored frames from the video stream.

4. **numpy:** It is used to perform transformations on muti-dimensional arrays.

- **expand_dims():** Expands image dimensions for use by keras functions.
- **around()** and **max()** are some other functions used.

5. **face_recognition:** It provides a series of functions to recognize and manipulate faces. It is the world’s simplest face recognition library, built using Dlib’s state-of-the-art face recognition and deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark. It is used to find faces in pictures, identify them and recognize who appears in each photo.

- **face_encodings():** It is used to encode face images as 128-d vectors.
- **compare_faces():** It is used to compare new encodings with existing ones for label matching.

6. **pickle:** it is used for serializing and de-serializing a Python object structures. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

- **dumps():** it saves serialized encodings to disk in a .pickle file.

7. **socket:** It provides functions for socket programming in Python.

- **socket():** It is used to create a socket.
- **bind():** It is used to bind a socket to a port number.
- **listen():** It is used to listen for incoming connections on a socket.
- **accept():** It is used to accept an incoming connection on the socket.
- **send():** It is used to send data back to source of socket connection.

- **stop():** It is used to destroy the socket.

5.4.2 PHP Functions

1. **Socket Functions:**

- **socket_create():** It creates a socket.
- **socket_connect():** It is used to establish a connection with a server.
- **socket_write():** It is used to send data into the socket.
- **socket_read():** It is used to receive data sent into the socket.
- **set_time_limit():** It is used to set the maximum timeout delay, in seconds.
- **socket_close():** It is used to close the socket.

2. **String Functions:**

- **strtoupper():** It converts all characters to uppercase.
- **strtolower():** It converts all characters to lowercase.
- **trim():**It removes whitespace characters from both sides of a string.
- **explode():**It breaks a string into an array.
- **preg_match():** It performs a regular expression match.

3. **File Functions:**

- **fopen():** It opens a file. It returns FALSE and an error on failure.
- **feof():** It checks if the "end-of-file" (EOF) has been reached and returns TRUE if an error occurs, otherwise it returns FALSE.
- **fgets():** It is used to return a line from an open file.
- **fclose():** It closes an open file. It returns TRUE on success or FALSE on failure.

4. **Array Functions:**

- **array():** It is used to create an array.

- **array_key_exists():** It checks an array for a specified key, and returns true if the key exists and false if the key does not exist.
- **array_push():** It inserts one or more elements to the end of an array.
- **array_search():** It searches an array for a value and returns the key.
- **isset():** It determines if a variable is declared and is different than NULL.

CHAPTER 6

TESTING

6.1 Introduction

Software testing is conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. It involves the execution of a software component or system component to evaluate one or more properties of interest (a program or application), with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

6.2 Unit testing

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

6.2.1 Unit test case 1

This test case checks to see if users has entered the name of a person or has left the field blank, as shown in Table 6.1 and Figure 6.1.

Table 6.1 Unit test case for Valid Input check

Sl No. of test case:	1
Name of test:	Input check
Item / Feature being tested:	Name field
Sample Input:	Blank field and click on Submit button.
Expected output:	Message ‘Please fill out this field’ is displayed.
Actual output:	Message ‘Please fill out this field’ is displayed.
Remarks:	Test succeeded

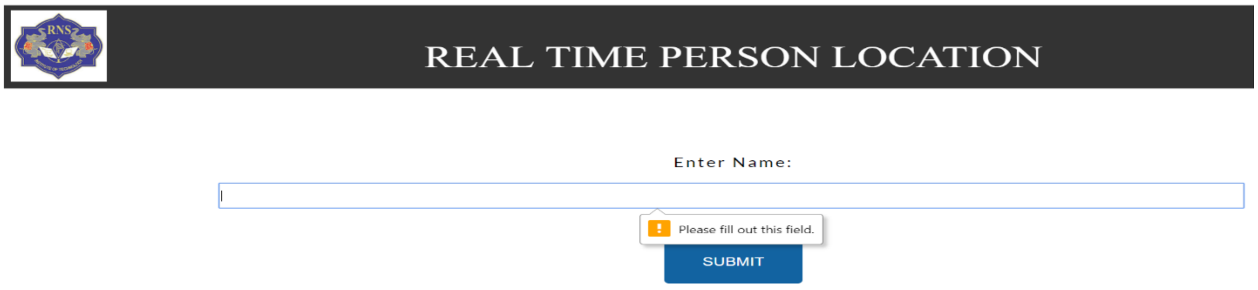


Figure 6.1 Unit test case for Valid Input check

6.2.2 Unit test case 2

This test case checks to see if users has entered the name of a person or has left the field blank, as shown in Table 6.2 and Figure 6.2.

Table 6.2 Unit test case for Invalid Input check

Sl No. of test case:	2
Name of test:	Input check
Item / Feature being tested:	Name field
Sample Input:	Name of Person to be Located and click on Submit button.
Expected output:	Message ‘Please fill out this field’ is not displayed.
Actual output:	Message ‘Please fill out this field’ is not displayed.
Remarks:	Test succeeded

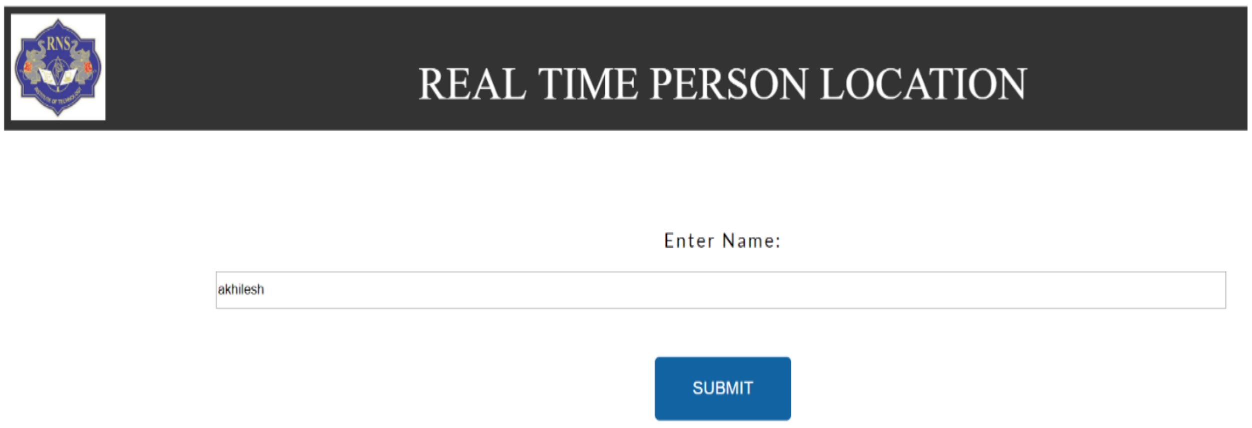


Figure 6.2 Unit test case for Invalid Input check

6.3 Integration testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

6.3.1 Integration test case 1

This test case checks to see if a known person is detected, correctly labelled and located in the right room(s), as shown in Table 6.3, and in Figure 6.3, 6.4 and 6.5.

Table 6.3 Integration test case for Known Face detection and Digit recognition check

Sl No. of test case:	1
Name of test:	Face detection and recognition, Digit recognition check
Item / Feature being tested:	Face detection and recognition, Digit recognition
Sample Input:	Frame with a Known Face and Frame with a Room Sign.
Expected output:	‘nikhilesh’ is located in Room 35.
Actual output:	‘nikhilesh’ is located in Room 35.
Remarks:	Test succeeded

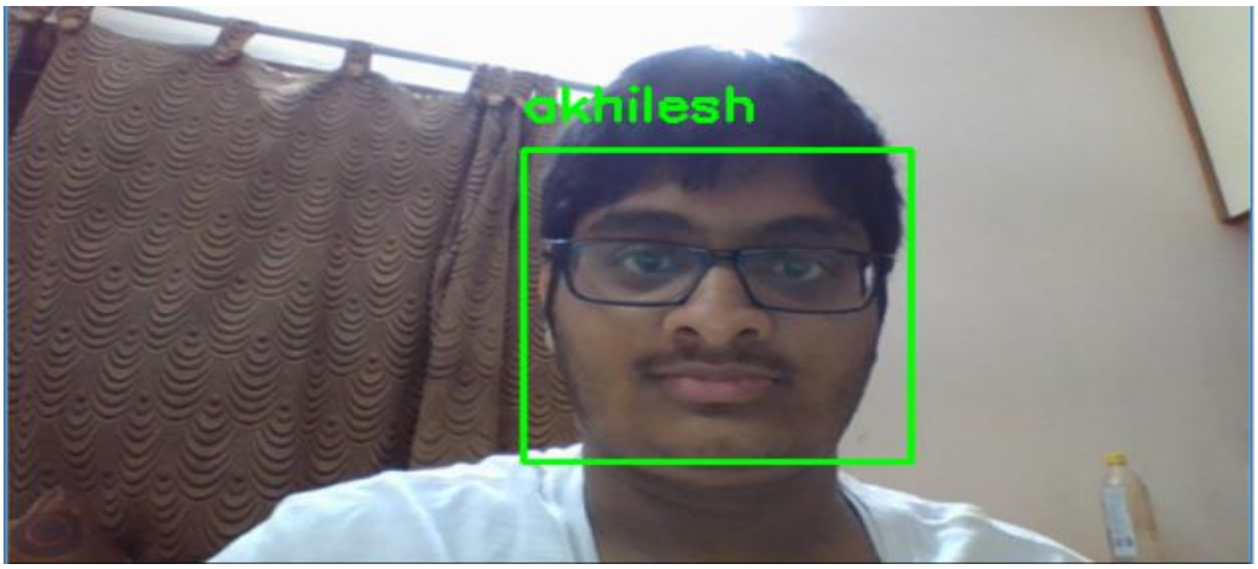


Figure 6.3 Unit test case for Known Face detection check

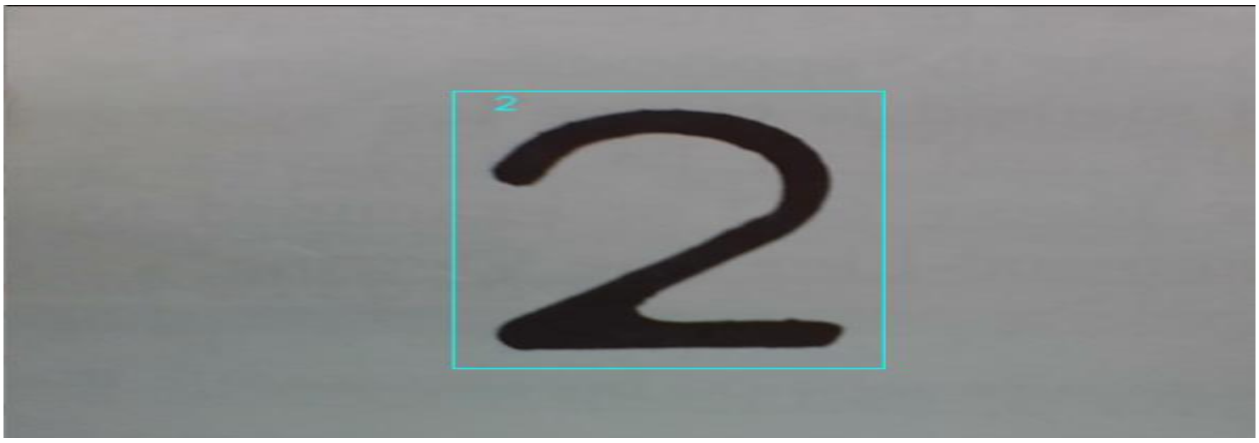
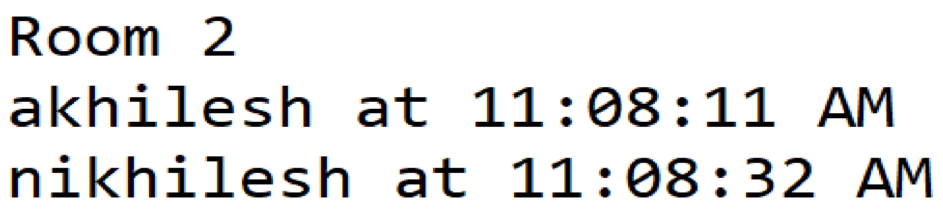


Figure 6.4 Unit test case for Digit recognition check



```
Room 2
akhilesh at 11:08:11 AM
nikhilesh at 11:08:32 AM
```

Figure 6.5 Unit test case for Digit extraction check

6.3.2 Integration test case 2

This test case checks to see if an unknown person is detected, correctly labelled and located in the right room(s), as shown in Table 6.4, and in Figure 6.6, 6.7 and 6.8.

Table 6.4 Integration test case for Unknown Face detection and Digits recognition check

Sl No. of test case:	2
Name of test:	Face detection and recognition, Digits recognition check
Item / Feature being tested:	Face detection and recognition, Digits recognition
Sample Input:	Frame with a Known Face and Frame with a Room Sign.
Expected output:	'Unknown Face 1' is located in Room 2357.
Actual output:	'Unknown Face 1' is located in Room 2357.
Remarks:	Test succeeded

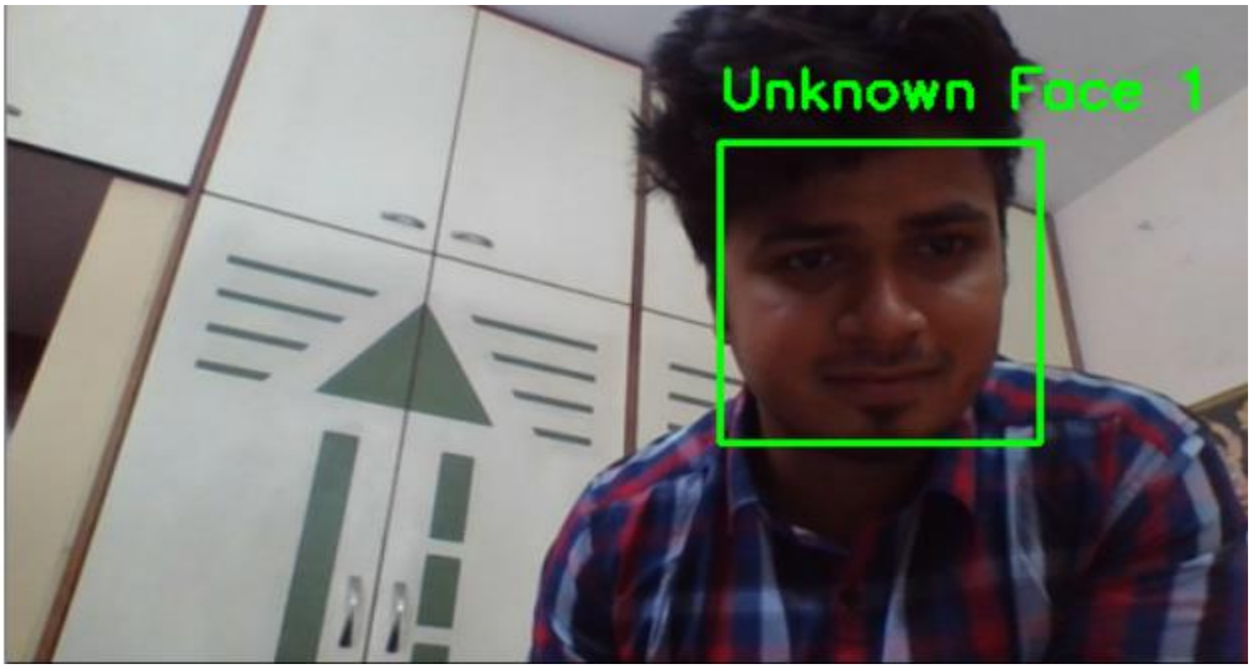


Figure 6.6 Unit test case for Unknown Face detection check

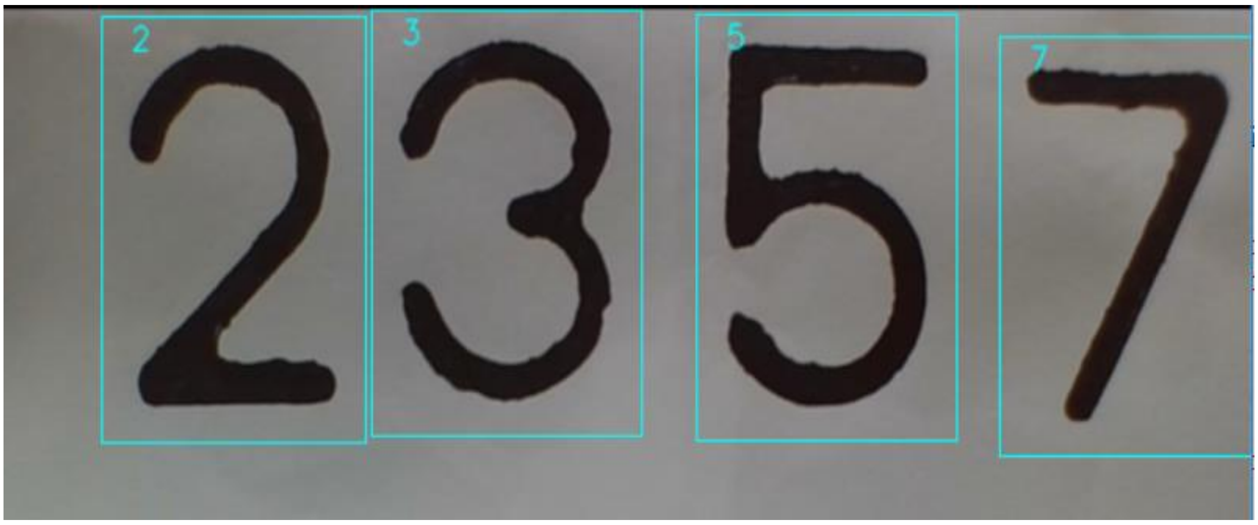
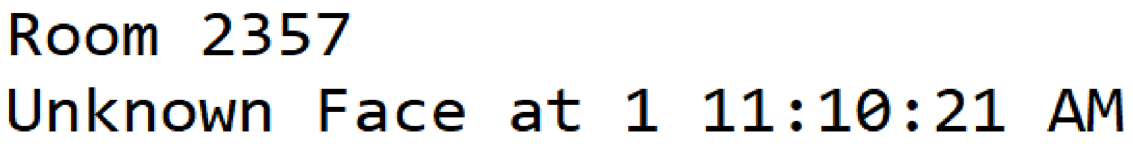


Figure 6.7 Unit test case for Digits recognition check



```
Room 2357
Unknown Face at 1 11:10:21 AM
```

Figure 6.8 Unit test case for Digits extraction check

6.4 System testing

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.

6.4.1 System test case 1

This test case checks if when the user has entered the name of a known person, the result is that the person has been found in some room, as shown in Table 6.5, and Figure 6.9, 6.10, 6.11, 6.12 and 6.13.

Table 6.5 System test case for Valid Input

Sl No. of test case:	1
Name of test:	Input check.
Items / Features being tested:	Input.
Sample Input:	Name of a Known Person.
Expected output:	Person is found in some room.
Actual output:	Person is found in some room
Remarks:	Test succeeded.

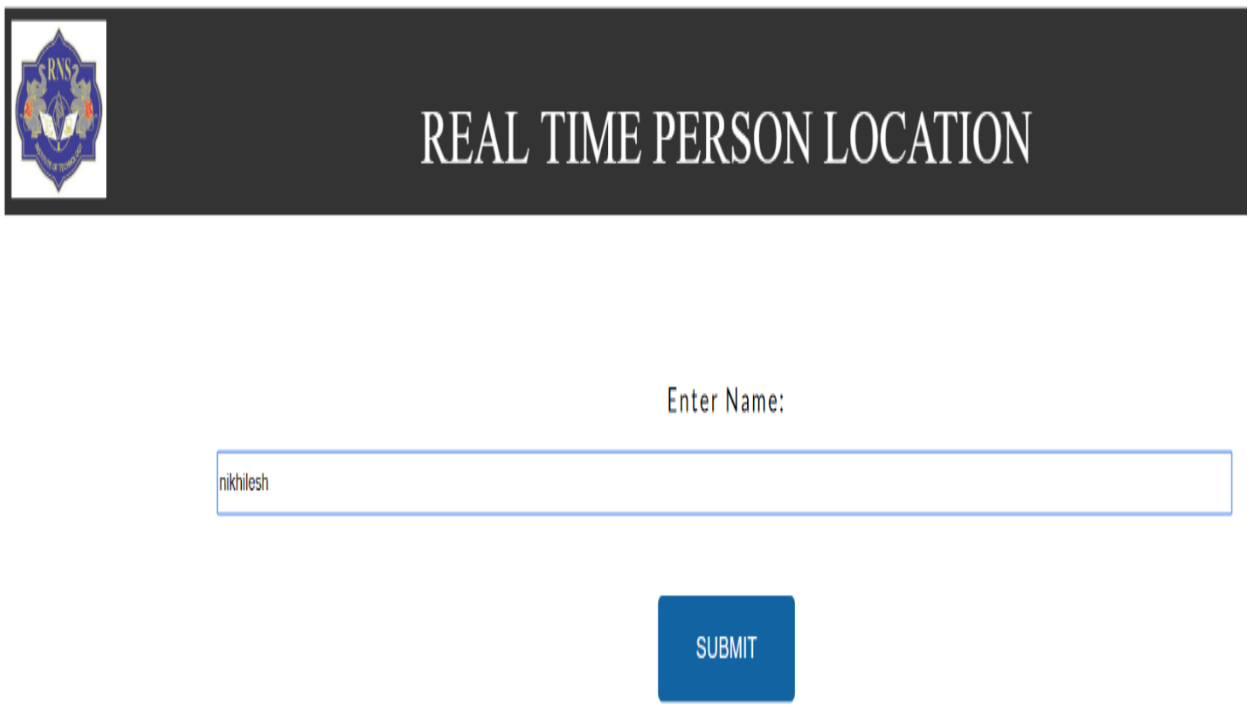


Figure 6.9 System test case for Valid Input

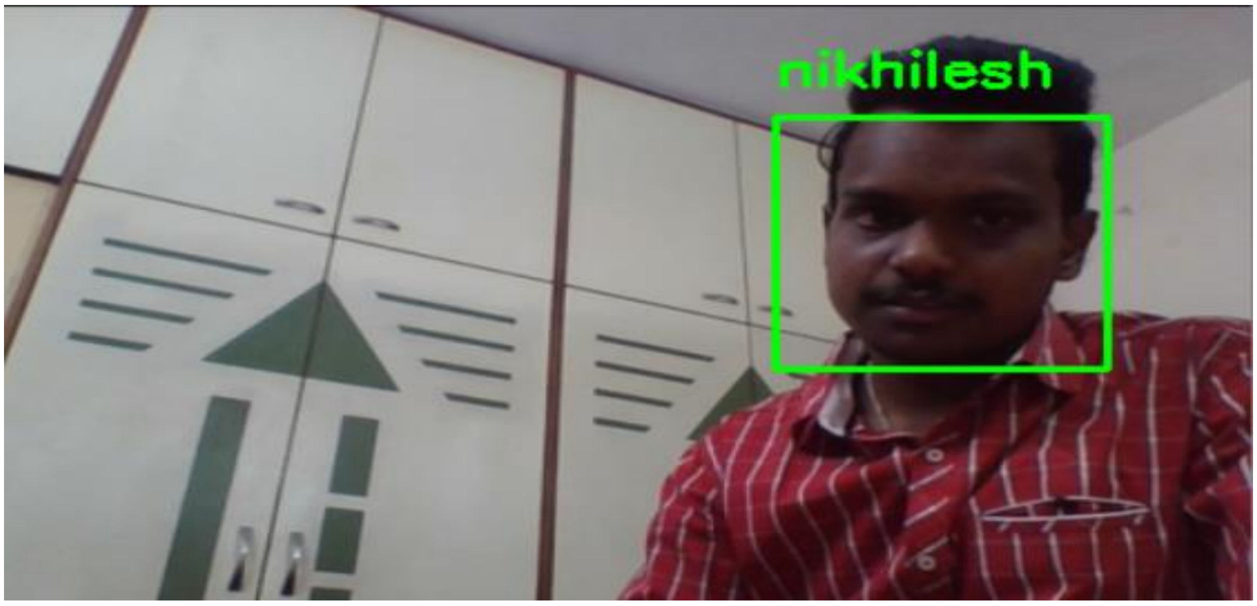


Figure 6.10 System test case for Known Face detection

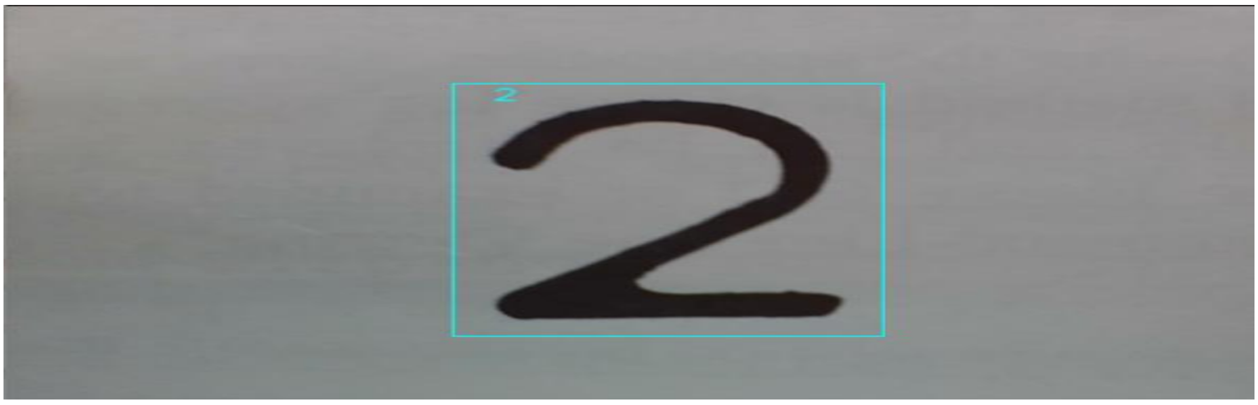
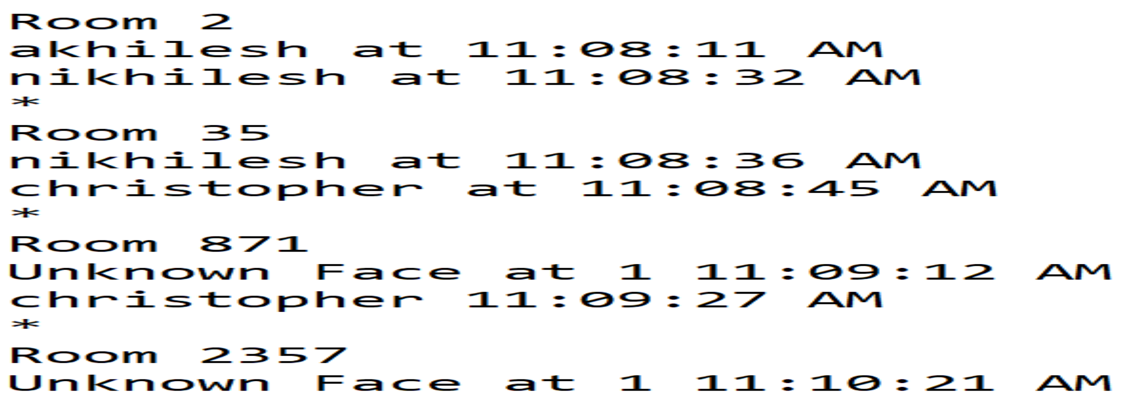
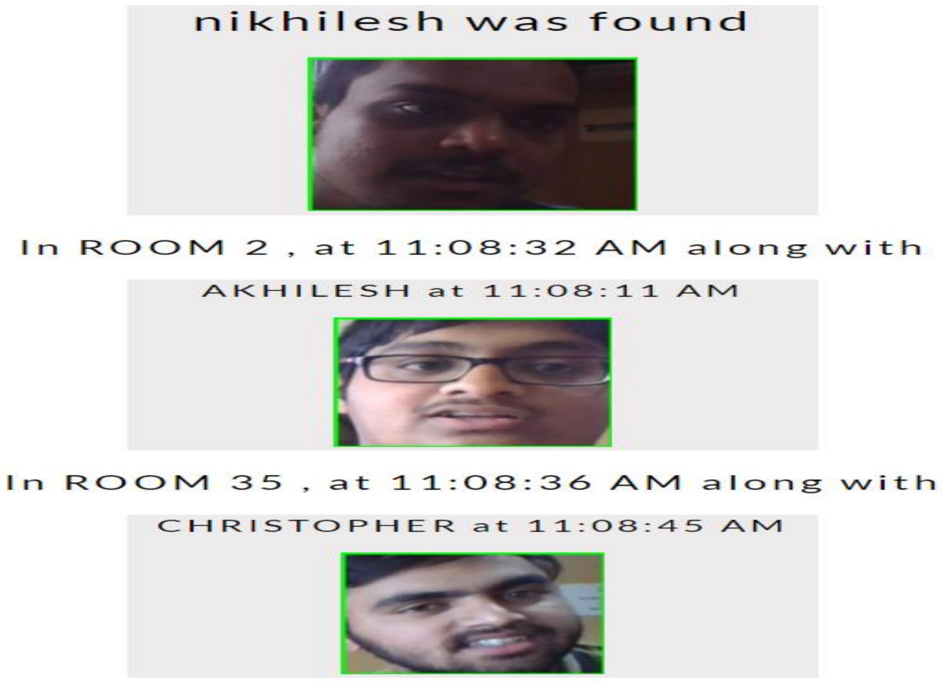


Figure 6.11 System test case for Digits extraction



```
Room 2
akhilesh at 11:08:11 AM
nikhilesh at 11:08:32 AM
*
Room 35
nikhilesh at 11:08:36 AM
christopher at 11:08:45 AM
*
Room 871
Unknown Face at 1 11:09:12 AM
christopher 11:09:27 AM
*
Room 2357
Unknown Face at 1 11:10:21 AM
```

Figure 6.12 System test case for Digits recognition


nikhilesh was found

In ROOM 2 , at 11:08:32 AM along with

AKHILESH at 11:08:11 AM

In ROOM 35 , at 11:08:36 AM along with

CHRISTOPHER at 11:08:45 AM

Figure 6.13 System test case for Output for Valid Input

6.4.2 System test case 2

This test case checks to see if when the user has entered the name of an unknown person, the result is that the person has not been found in any room, as shown in Table 6.6, and Figure 6.14, 6.15 and 6.16.

Table 6.6 System test case for Invalid Input

Sl No. of test case:	2
Name of test:	Input check.
Items / Features being tested:	Input.
Sample Input:	Name of an Unknown Person.
Expected output:	Person is not found in any room.
Actual output:	Person is not found in any room
Remarks:	Test succeeded.

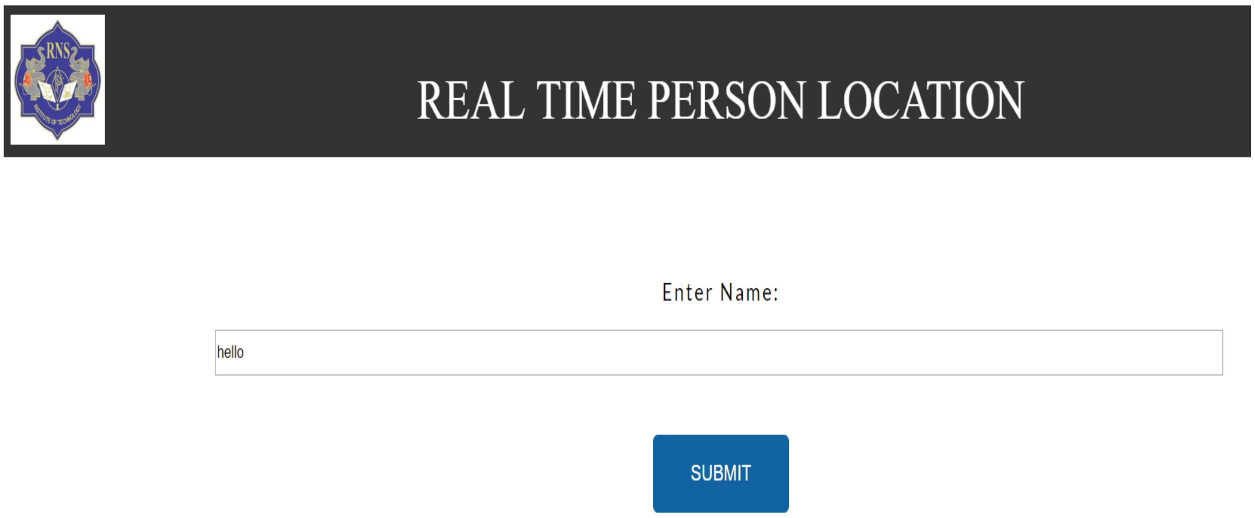


Figure 6.14 System test case for Invalid Input

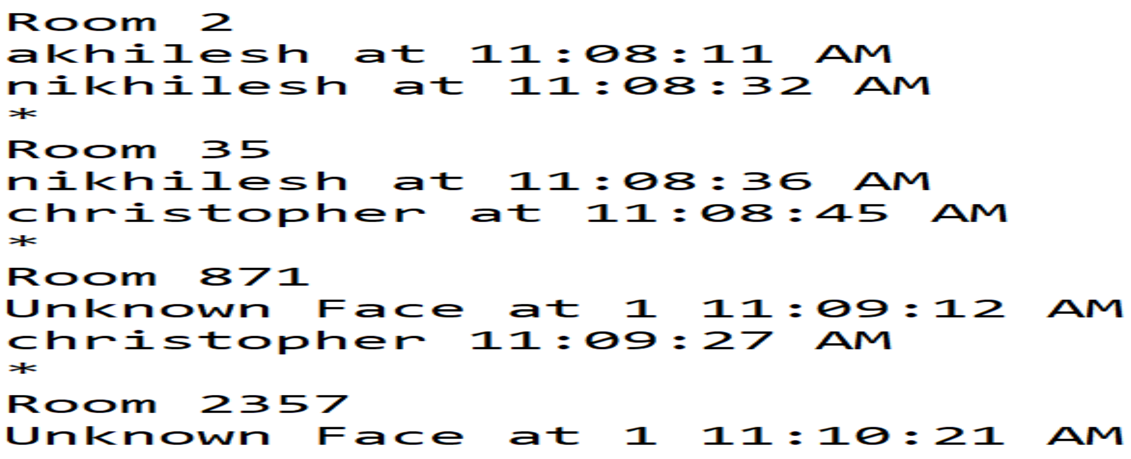
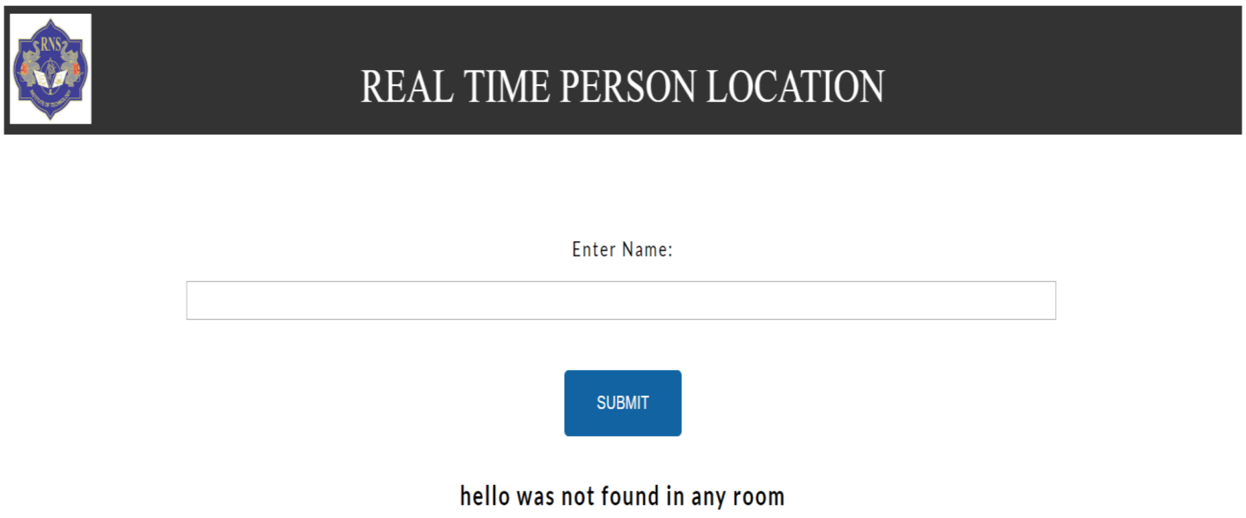

```
Room 2
akhilesh at 11:08:11 AM
nikhilesh at 11:08:32 AM
*
Room 35
nikhilesh at 11:08:36 AM
christopher at 11:08:45 AM
*
Room 871
Unknown Face at 1 11:09:12 AM
christopher 11:09:27 AM
*
Room 2357
Unknown Face at 1 11:10:21 AM
```

Figure 6.15 System test case for System output



hello was not found in any room

Figure 6.16 System test case for Output for Invalid Input

6.5 Validation Testing

Validation Testing is the process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. It ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

6.5.1 Validation test case 1

This test case checks to see if the system detects the face of the given name in the correct room(s) or not, as shown in Table 6.7, and Figure 6.17 and 6.18.

Table 6.7 Validation test case for Valid Input

Sl No. of test case:	1
Name of test:	Face detection and recognition check
Item / Feature being tested:	Face detection and recognition
Sample Input:	Name ‘akhilesh’.
Expected output:	‘akhilesh was found’ is displayed.
Actual output:	‘akhilesh was found’ is displayed.
Remarks:	Test succeeded

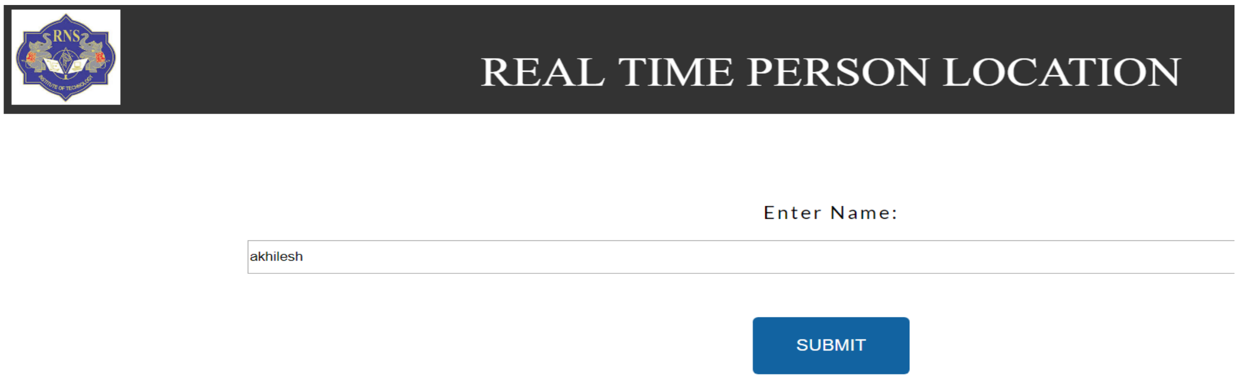
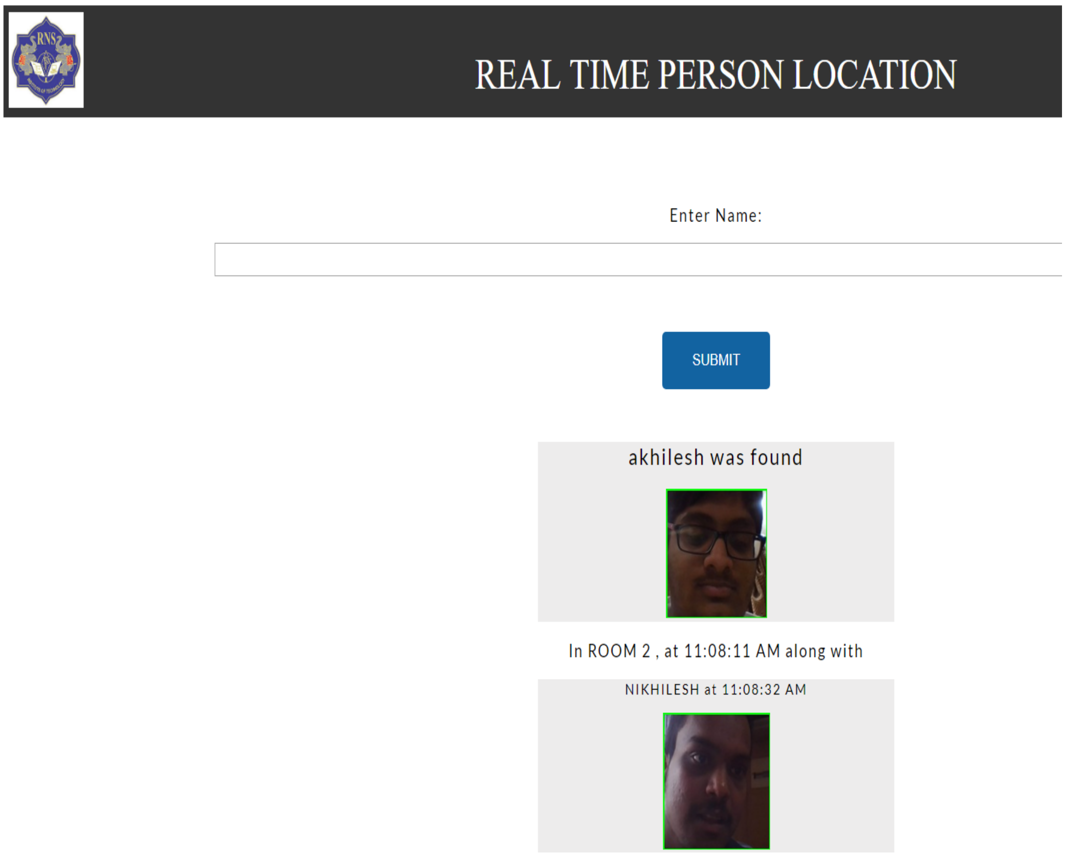


Figure 6.17 Valid Input



akhilesh was found

In ROOM 2 , at 11:08:11 AM along with

NIKHILESH at 11:08:32 AM

Figure 6.18 Output for Valid Input

6.5.2 Validation test case 2

This test case checks to see if the system does not detect the face of the given invalid name in any room(s) or not, as shown in Table 6.8, and Figure 6.19 and 6.20.

Table 6.8 Validation test case for Invalid Input

Sl No. of test case:	2
Name of test:	Face detection and recognition check
Item / Feature being tested:	Face detection and recognition
Sample Input:	Name ‘abcd’.
Expected output:	‘abcd was not found in any room’ is displayed.
Actual output:	‘abcd was not found in any room’ is displayed.
Remarks:	Test succeeded

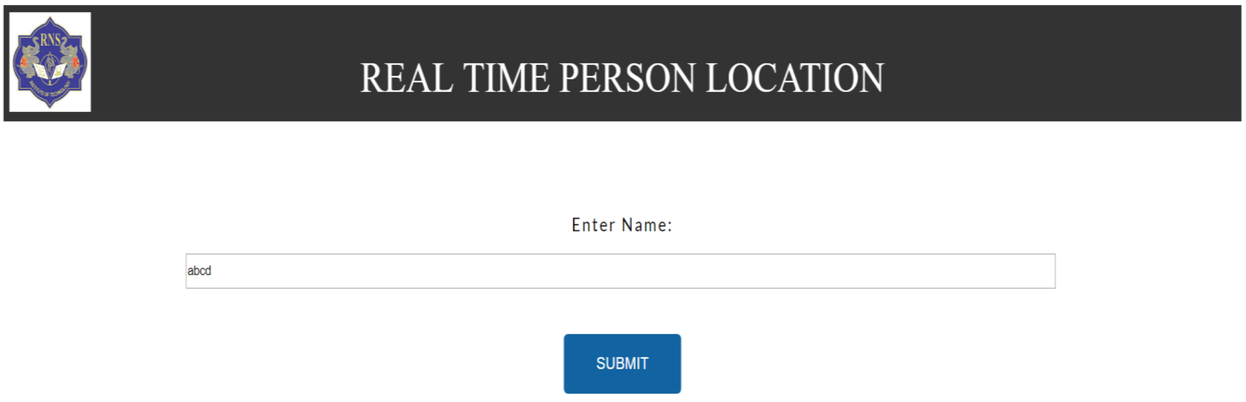
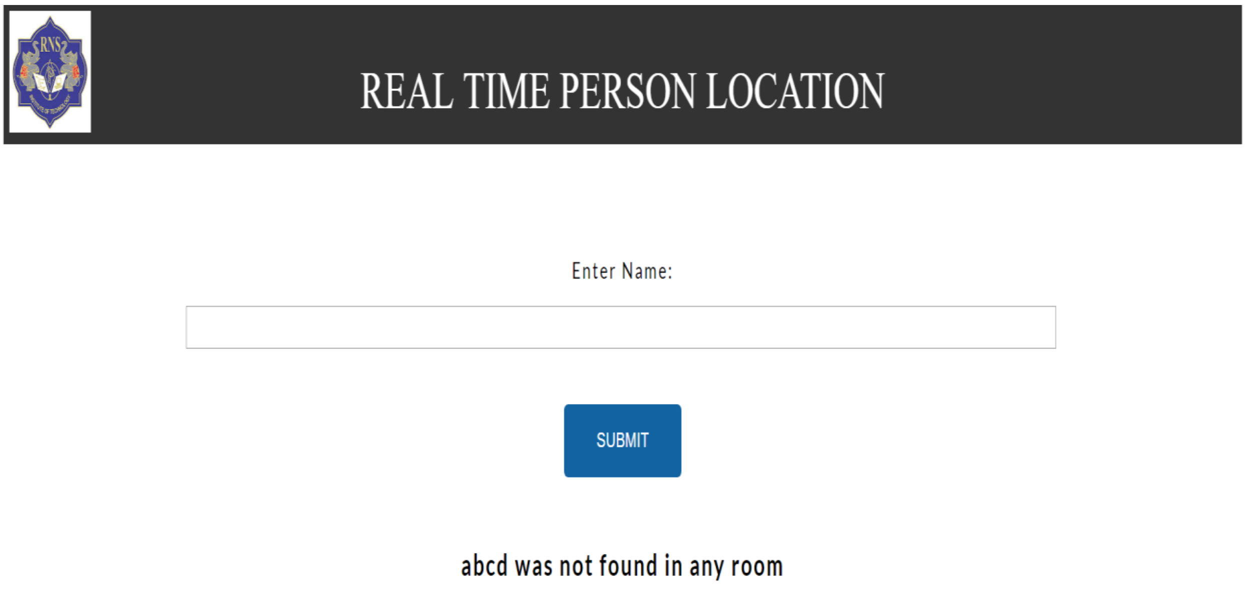


Figure 6.19 Validation test case for Invalid Input



abcd was not found in any room

Figure 6.20 Output for Invalid Input

6.6 User Acceptance Testing

User acceptance testing (UAT) consists of a process of verifying that a solution works for the user. It is not system testing, but rather ensures that the solution will work for the user. User acceptance testing is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications.

6.6.1 User Acceptance test case 1

This test case checks to see if the correct label for the face is detected in the image or not, as shown in Table 6.9 and Figure 6.21.

Table 6.9 User Acceptance test case for Known Face Detection

Sl No. of test case:	1
Name of test:	Face detection and recognition check
Item / Feature being tested:	Face detection and recognition
Sample Input:	Frame with a Known Face.
Expected output:	Label ‘christopher’ is displayed.
Actual output:	Label ‘christopher’ is displayed.
Remarks:	Test succeeded

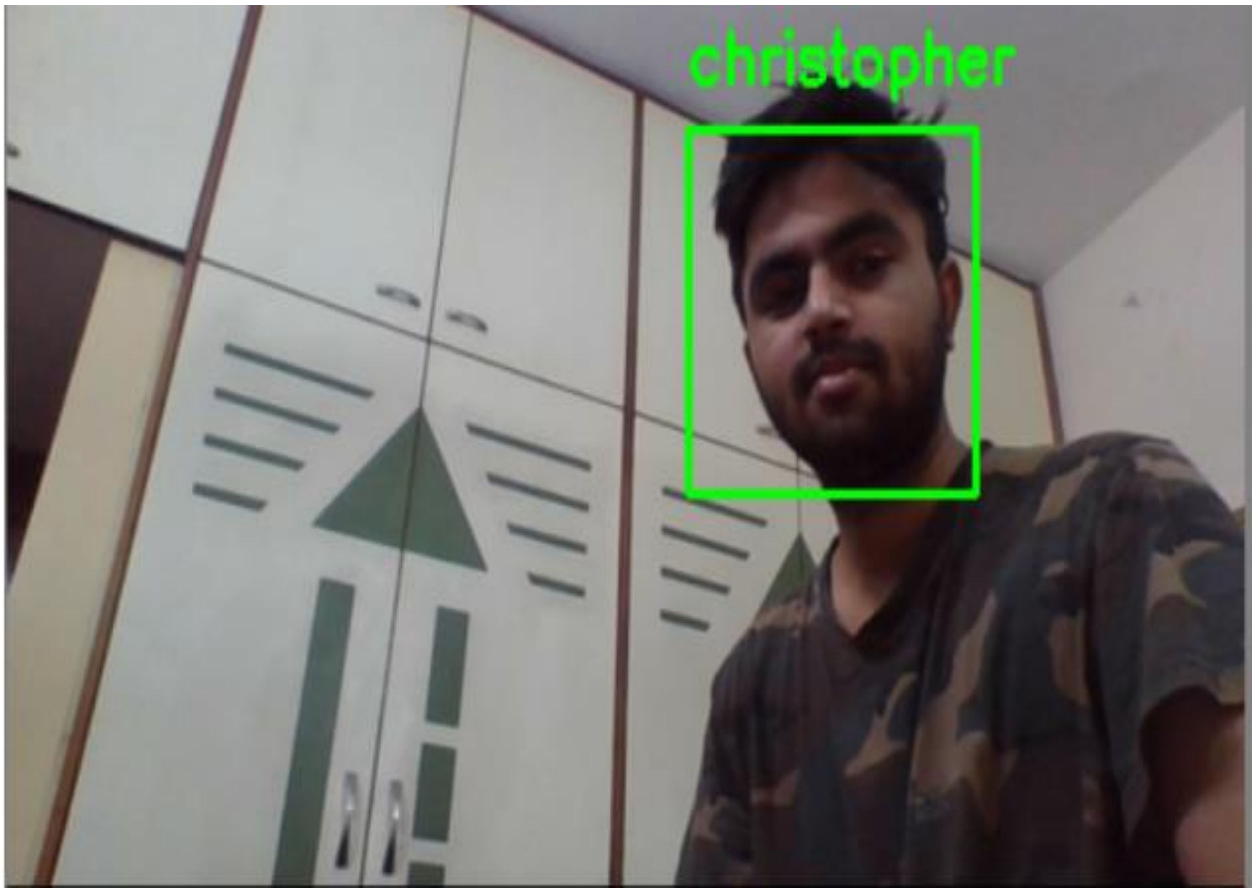


Figure 6.21 User Acceptance test case for Known Face Detection

6.6.2 Unit User Acceptance test case 2

This test case checks to see if the correct label for the face is detected in the image or not, as shown in Table 6.10 and Figure 6.22.

Table 6.10 User Acceptance case for Unknown Face Detection

Sl No. of test case:	2
Name of test:	Face detection and recognition check
Item / Feature being tested:	Face detection and recognition
Sample Input:	Frame with an Unknown Face.
Expected output:	Label 'Unknown Face' is displayed.
Actual output:	Label 'Unknown Face' is displayed.
Remarks:	Test succeeded

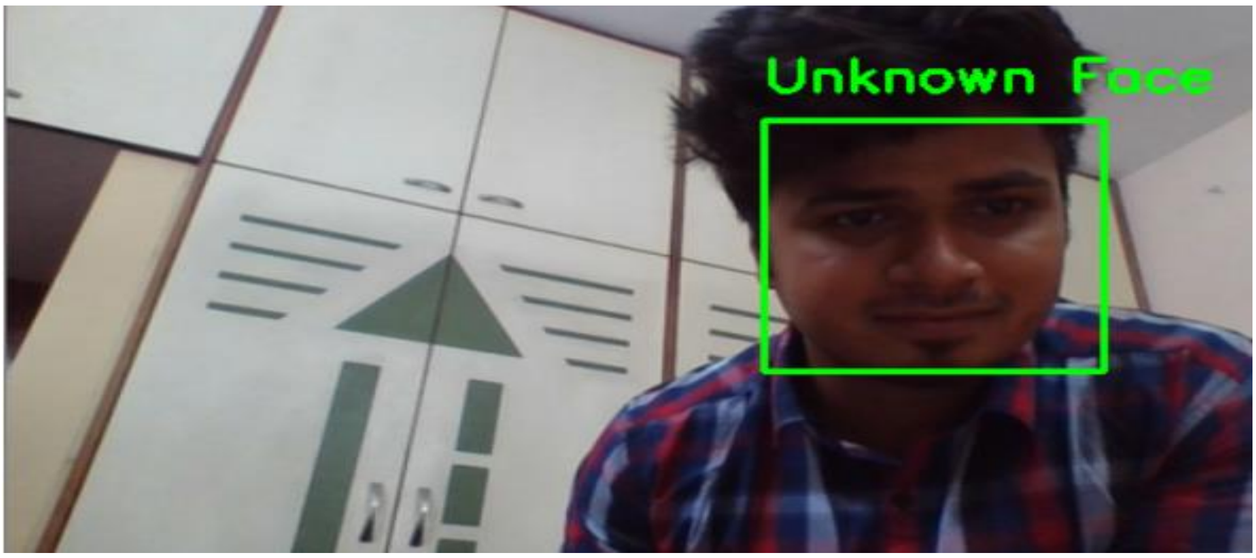


Figure 6.22 User Acceptance test case for Unknown Face Detection

CHAPTER 7

DISCUSSION OF RESULTS

The outcomes of test results for a variety of user interactions with the application are discussed in the following section of the chapter.

7.1 Home page

Figure 7.1 shows the Home page of the Person Location system that is displayed to the users.

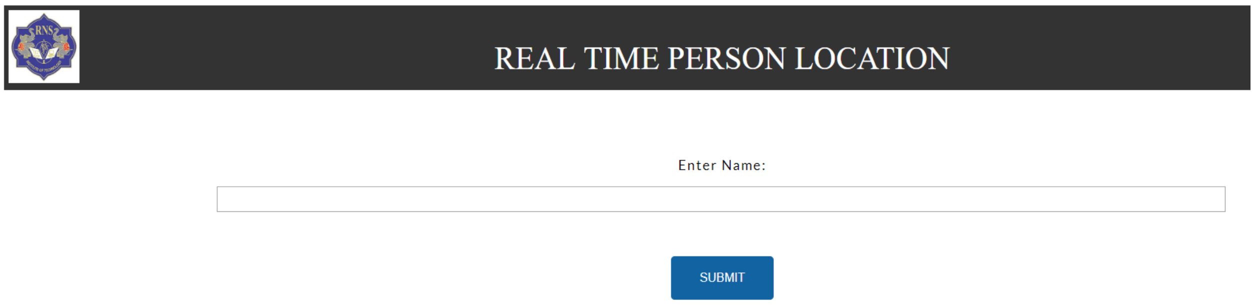
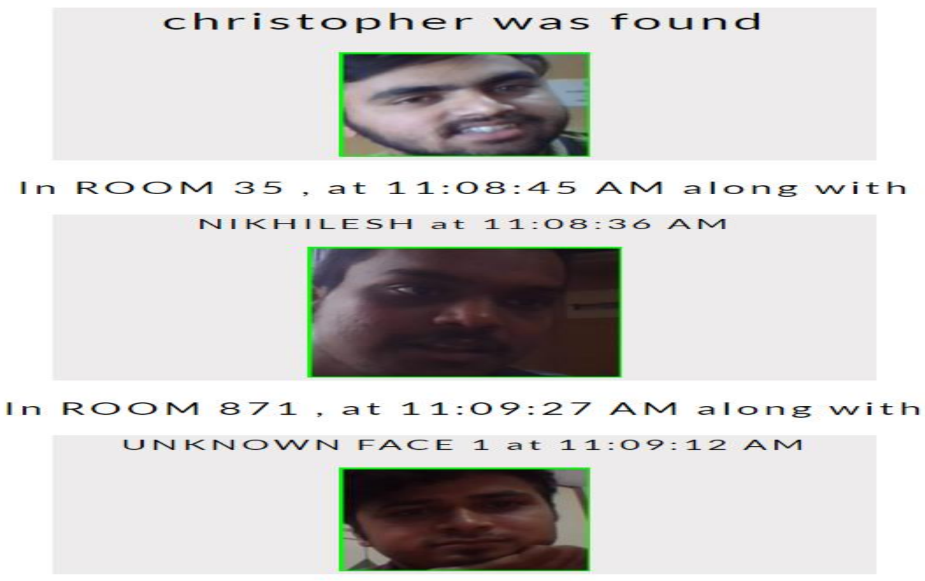


Figure 7.1 Home page

7.2 Result page – Person Located

Figure 7.2 shows the result page when the person whose name has been entered is found. The page displays a sample frame in which the person's face was detected. It shows the rooms in which the person was located and along with the names of the other occupants of the room when the person was found.



christopher was found

In ROOM 35 , at 11:08:45 AM along with

NIKHILESH at 11:08:36 AM

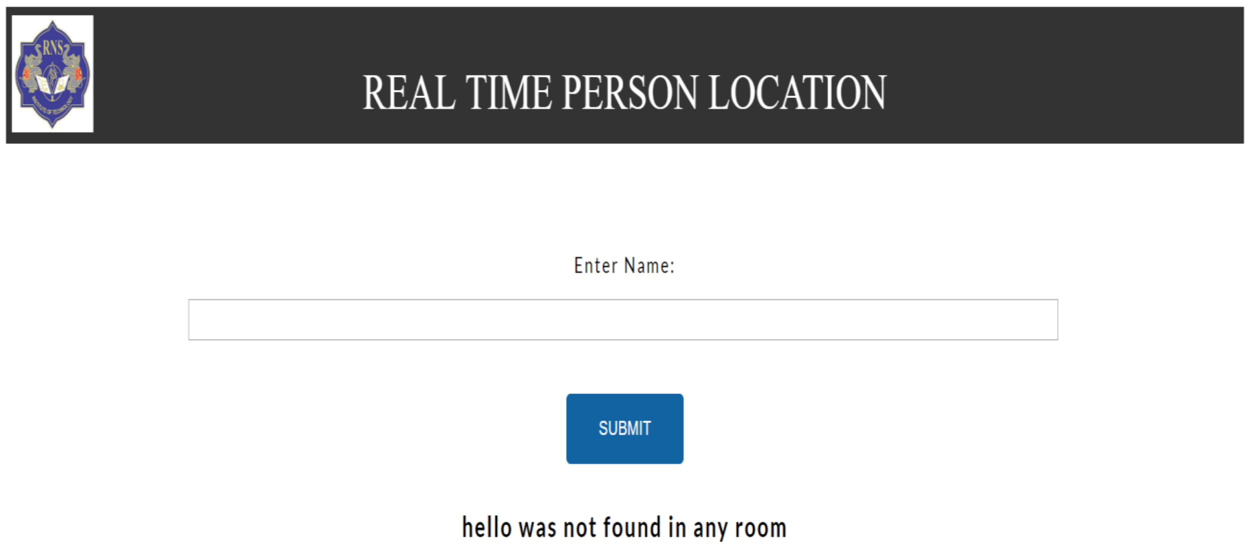
In ROOM 871 , at 11:09:27 AM along with

UNKNOWN FACE 1 at 11:09:12 AM

Figure 7.2 Result page – Person Located

7.3 Result page – Person not Located

Figure 7.3 shows the result page when the person whose name has been entered is not found.

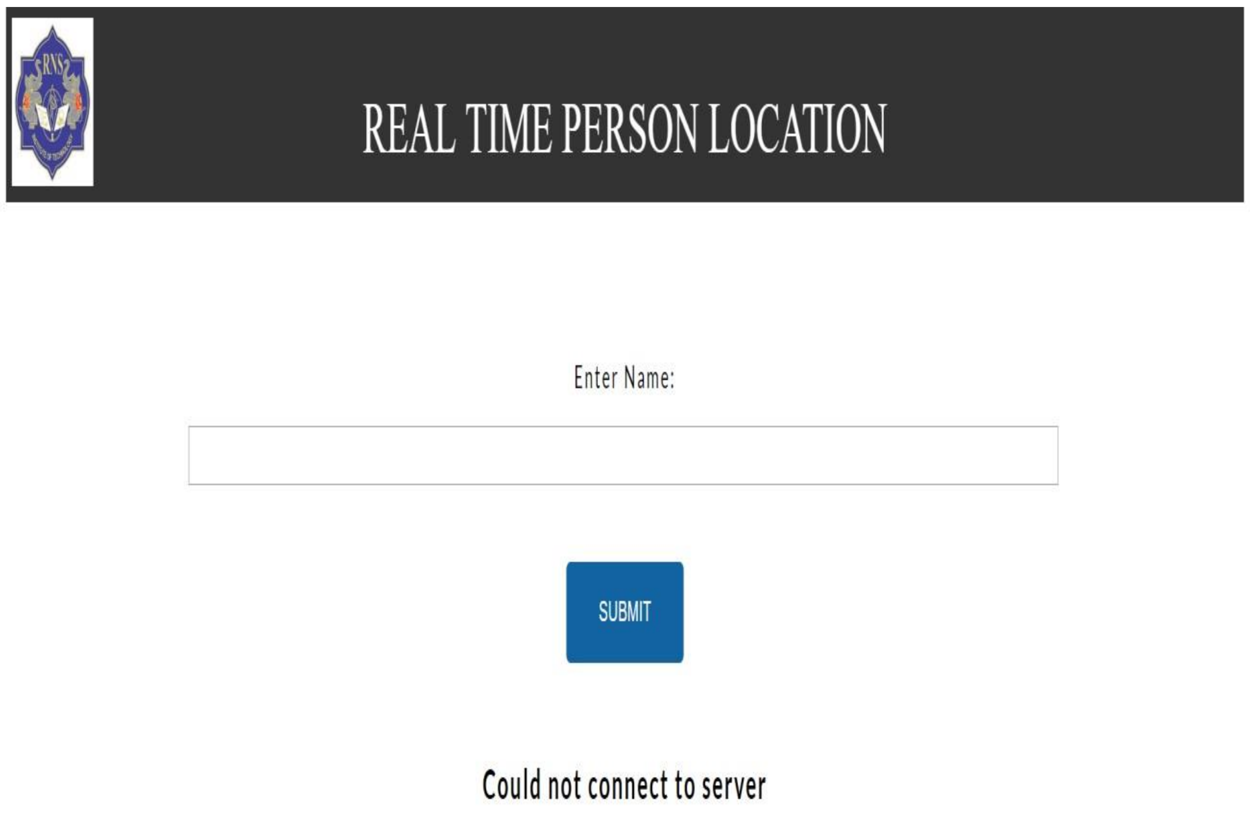


hello was not found in any room

Figure 7.3 Result page – Person not Located

7.4 Result page – Server not accepting connections

Figure 7.4 shows the appropriate message is displayed if the server is not accepting connections.

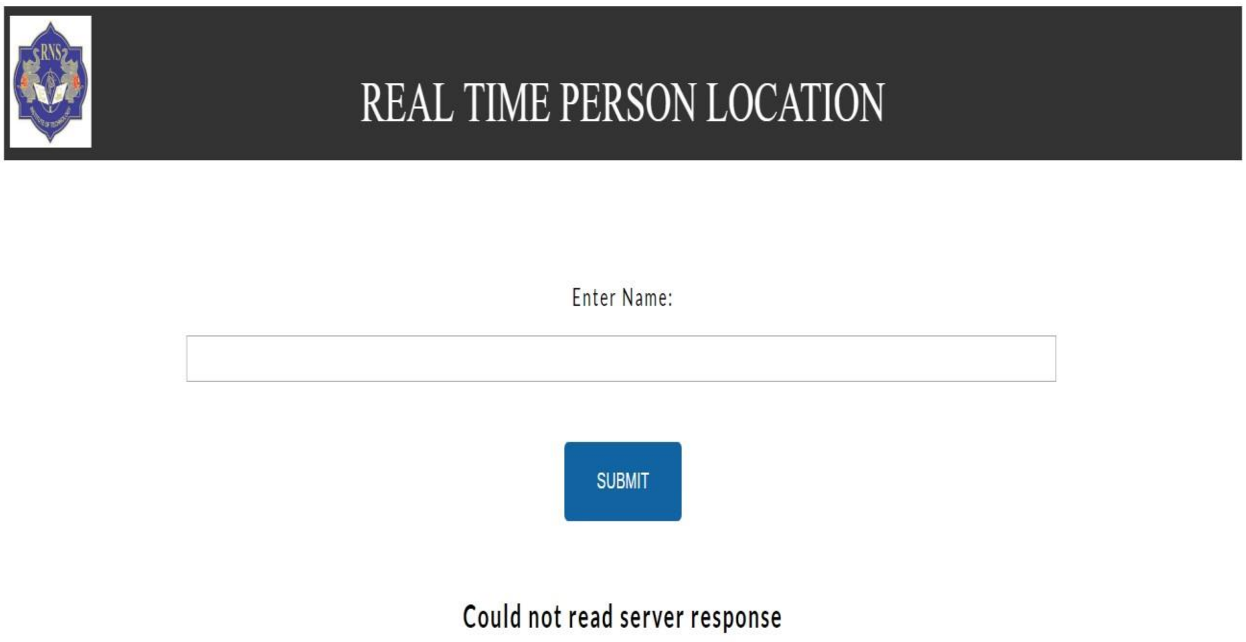


Could not connect to server

Figure 7.4 Result page – Server not accepting connections

7.5 Result page – Server fails

Figure 7.5 shows the appropriate message is displayed if the server fails.

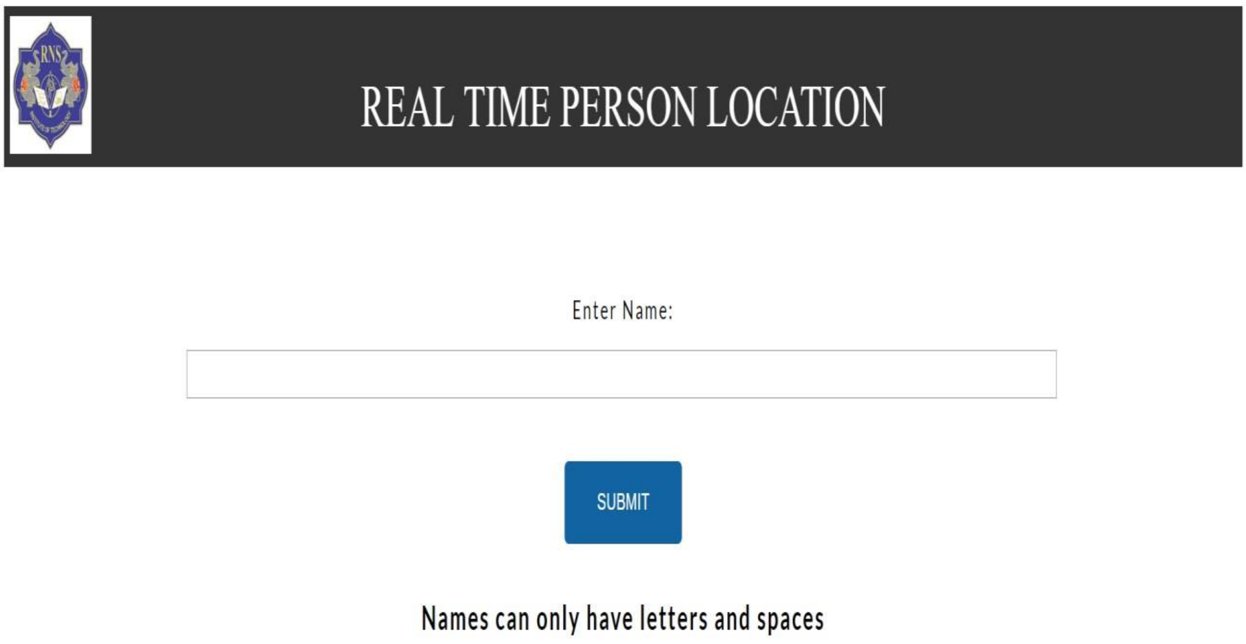


Could not read server response

Figure 7.5 Result page – Server fails

7.6 Result page – Name contains special characters

Figure 7.6 shows the appropriate message is displayed if the input name contains special characters.



Names can only have letters and spaces

Figure 7.6 Result page – Name contains special characters

7.7 Two digits are recognized and extracted

Figure 7.7 and 7.8 shows a two digits are recognized and extracted from the image.

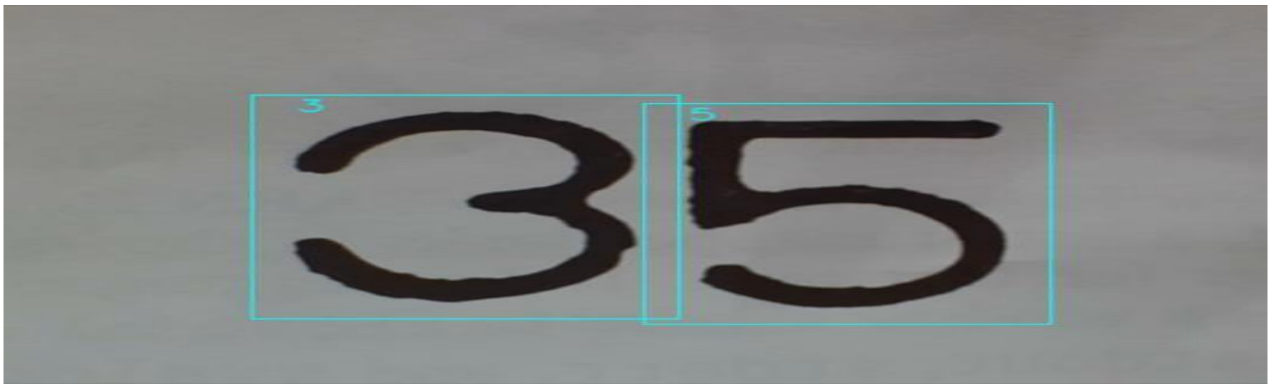


Figure 7.7 Two digits are recognized

Room 35

Figure 7.8 Two digits are extracted

7.8 Three digits are recognized and extracted

Figure 7.9 and 7.10 shows a three digits are extracted from the image.

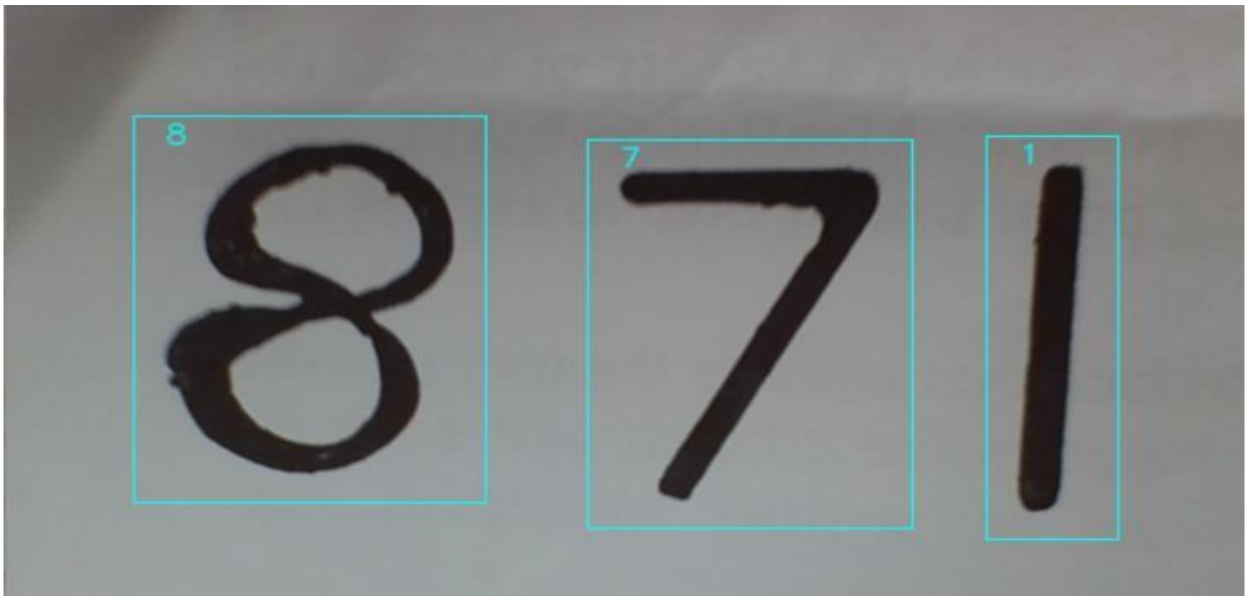


Figure 7.9 Three digits are recognized

Room 871

Figure 7.10 Three digits are extracted

7.9 Known Face Detection

Figure 7.11 shows the detection of a person’s face in the real-time video stream. A bounding box is displayed around the person’s face, and the name of the person is displayed as a label at the top left corner, on top of the box. 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.

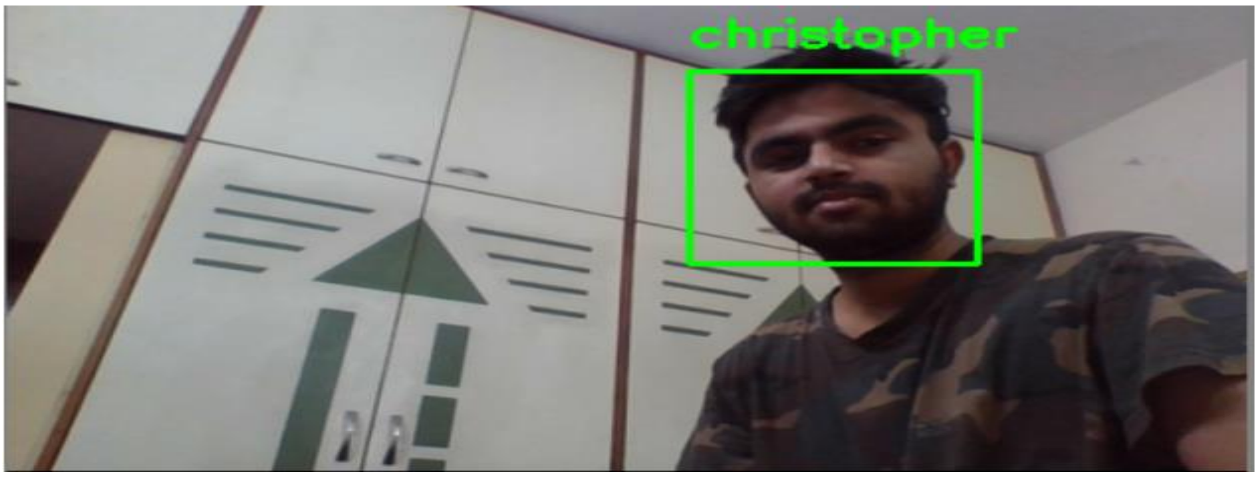


Figure 7.11 Known Face Detection

7.10 Unknown Face Detection – Before Retraining CNN

Figure 7.12 shows the detection of a person’s face in the real-time video stream. A bounding box is displayed around the unknown person’s face, and ‘Unknown Face’ is displayed as a label at the top left corner, on top of the box.

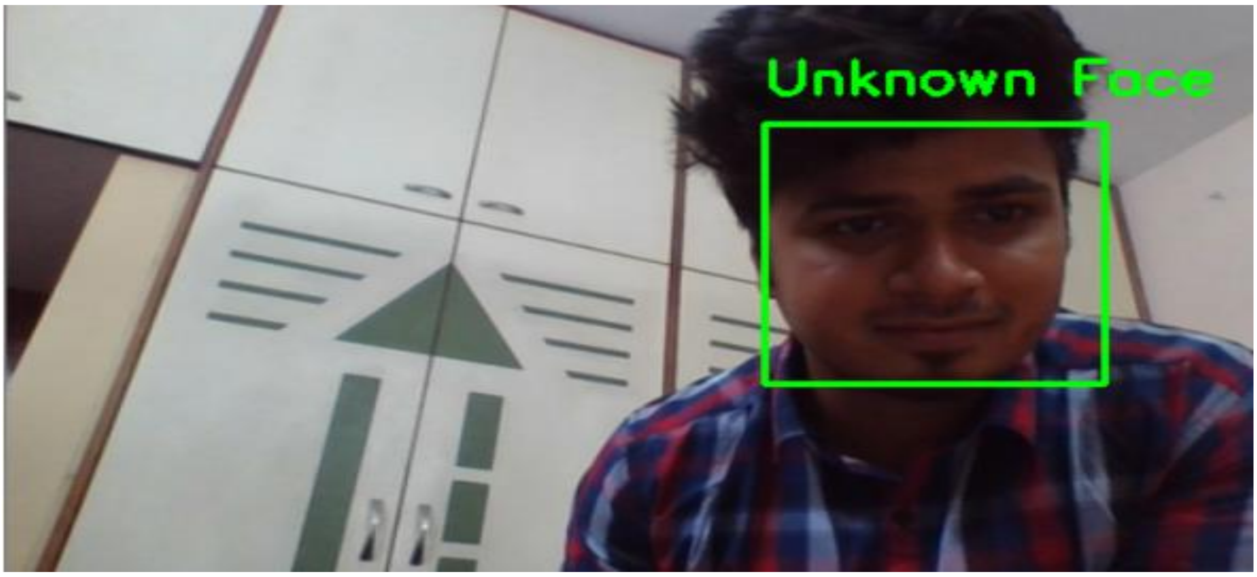


Figure 7.12 Unknown Face Detection – Before Retraining CNN

7.11 Results of Unknown Face Detection

Figure 7.13 shows the storing of newly captured images of detected Unknown Face.

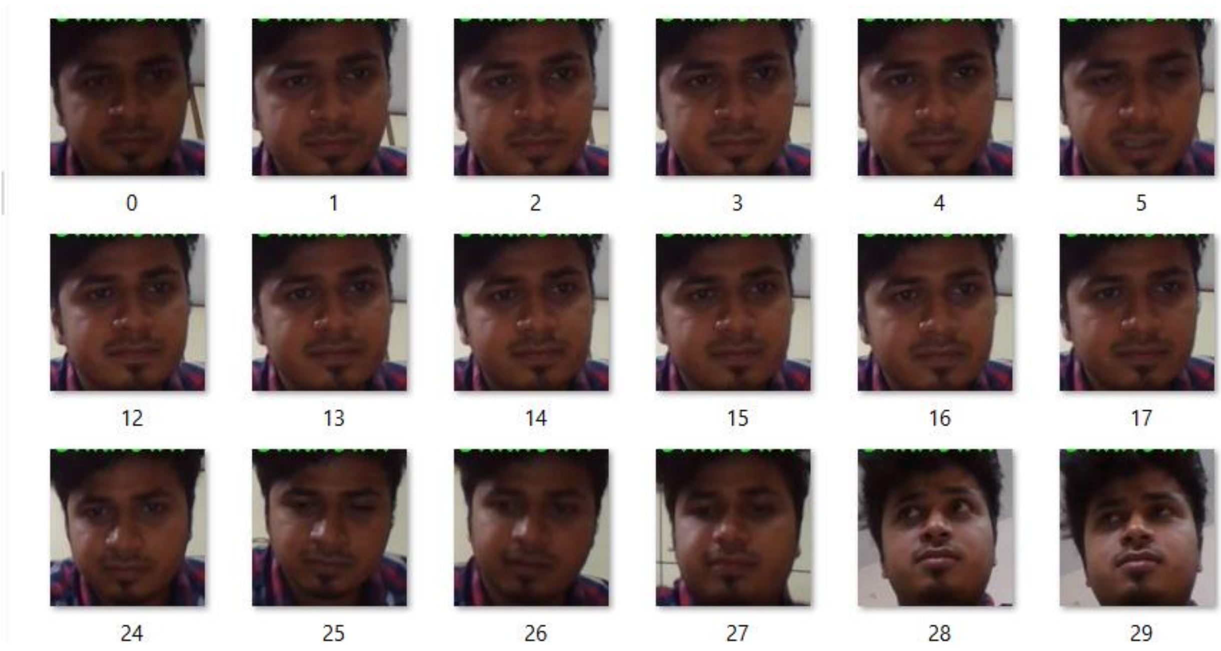


Figure 7.13 Results of Unknown Face Detection

7.12 Unknown Face Detection – After Retraining CNN

Figure 7.14 shows the detection of a person’s face in the real-time video stream, after CNN is retrained on new images. A bounding box is displayed around the unknown person’s face, and ‘Unknown Face 1’ is displayed as a label at the top left corner, on top of the box.

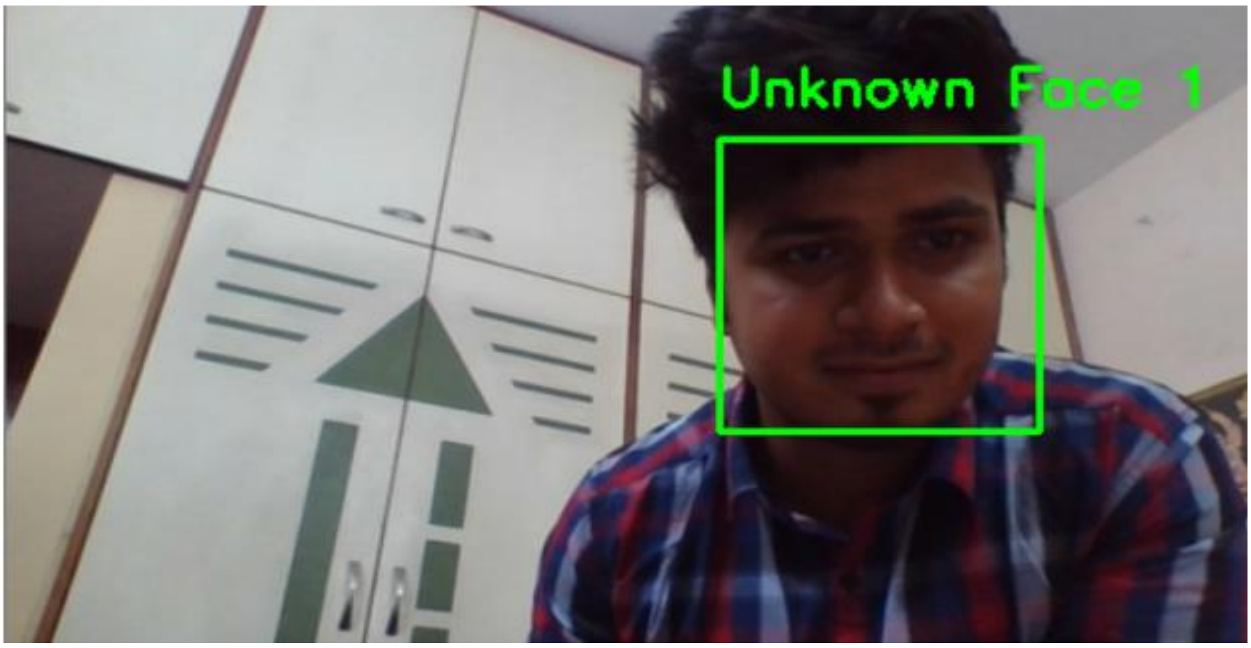
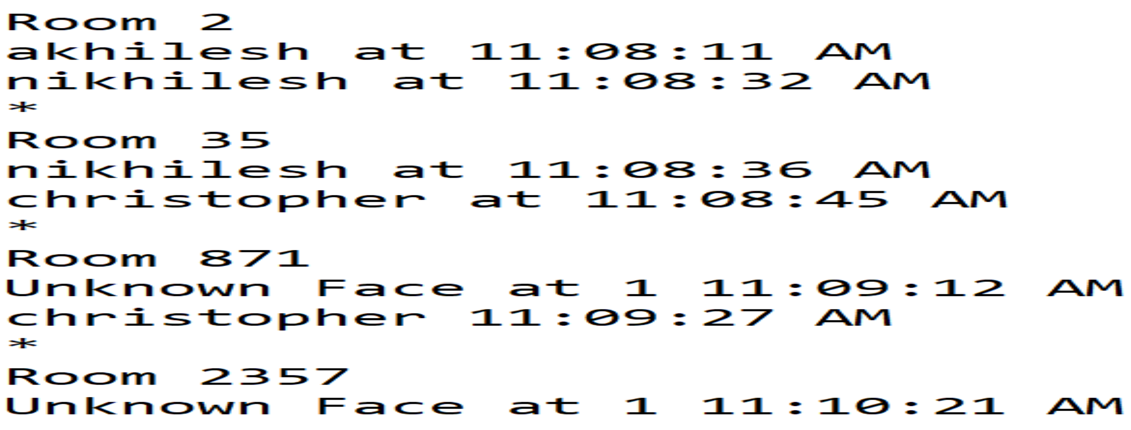


Figure 7.14 Unknown Face Detection – After Retraining CNN

7.13 Results of Face Detection + Text Recognition

Figure 7.15 shows the results after the video stream has been terminated.



```
Room 2
akhilesh at 11:08:11 AM
nikhilesh at 11:08:32 AM
*
Room 35
nikhilesh at 11:08:36 AM
christopher at 11:08:45 AM
*
Room 871
Unknown Face at 1 11:09:12 AM
christopher 11:09:27 AM
*
Room 2357
Unknown Face at 1 11:10:21 AM
```

Figure 7.15 Results of Face Detection + Text Recognition

CHAPTER 8

CONCLUSION AND FUTURE WORK

The Real-Time Person Location System performs Video Content Analysis on the live footage being streamed from a camera. Text extraction and recognition is performed on the Location tags made visible in the video's frames. A Deep, Convolutional Neural Network is trained on a dataset of face images and is used to perform Facial Recognition on the camera footage of a room of sitting occupants. The system is able to detect and locate a person with good precision and accuracy.

The Image Processing involved during the operation of the system uses different libraries that have been made available as Open Source. Each library helps in improving the quality of facial recognition, by optimizing different aspects of the processing techniques. Python libraries are used to generate augmented images. This has made the system robust and it is accurate with a considerably good performance on images having complex background, detecting faces of various sizes, postures and expressions, even under uncontrolled lighting environments.

The system is useful for various security and attendance purposes, where it will reduce the need for interaction with the user/person for recognition and authentication, as well as prevent any user from successfully imitating another person.

As a future enhancement, the dataset and outputs can be stored in and retrieved from a database. A wireless camera can be used to improve mobility of video stream. The system can be further enhanced to a network of surveillance cameras. Training of persons can be done as a background process and the system shall be functional on the foreground.

REFERENCES

[1] Efim V. Zatonskikh, Georgii I. Borzunov, Konstantin Kogos," Development of Elements of Two-Level Biometric Protection Based on Face and Speech Recognition in the Video Stream", 2018

[2] Sergey A. Alyamkin, Nikita A. Nikolenko, Vladimir V. Dyubanov, Evgeniy N. Pavlovskiy, "FRiS-censoring of reference sample in face recognition task by deep neural networks", 2017

[3] Sheshang Degadwala, Vidisha Patel, Sagar Pandya, Shraddha Shah, Udita Doshi," A review on real time face tracking and identification for surveillance system", 2016

[4] Yikui Zhai, Hui Ma, Ying Xu, "Pose over complete automatic registration method for video based robust face recognition", 2017

[5] Aayush Mittal, Fatima Sartaj Khan, Praveen Kumar, Tanupriya Choudhury, "Cloud Based Intelligent Attendance System through Video Streaming", 2017

[6] Injae Lee, Heechul Jung, Chung Hyun Ahn, Jeongil Seo, Junmo Kim, Ohseok Kwon," Real-time Personalized Facial Expression Recognition System Based on Deep Learning", 2016

[7] Rosali Mohanty, M.V. Raghunadh," A New Approach to Face Detection based on YCgCr Color Model and Improved AdaBoost Algorithm", 2016

[8] Enhao Guan, Zhaohui Zhang, Mengzhong He, Xiaoyan Zhao," Evaluation of classroom teaching quality based on video processing technology", 2017

[9] Marko Arsenovic, Srdjan Sladojevic, Andras Anderla, Darko Stefanovic," FaceTime – Deep Learning Based Face Recognition Attendance System", 2017

[10] Seyed Ali Miraftabzadeh, Paul Rad, Kim-Kwang Raymond Choo and Mo Jamshidi," A Privacy-Aware Architecture at the Edge for Autonomous Real-Time Identity Reidentification in Crowds", 2017

[11] Deng-Yuan Huang, Chao-Ho Chen, Tsong-Yi Chen, Jian-He Wu, Chien-Chuan Ko, "Real-Time Face Detection Using a Moving Camera", 2018

[12] Chenghao Zheng, Menglong Yang, Chengpeng Wang, "A real-time face detector based on an end-to-end CNN", 2017

[13] Jianchao Li, Dongping Zhang, Kun Zhang, Kui Hu, Li Yang, "Real-time Face Detection During the Night", 2017

[14] Jiajun Wang, Beizhan Wang, Yinhuan Zheng, Weiqiang, Liu, "Research and Implementation on Face Detection Approach Based on Cascaded Convolutional Neural Networks", 2017

[15] Goutham Reddy Kotapalle, Sachin Kotni," Security using Image Processing and Deep Convolutional Neural Networks", 2018