A Project Report on

Smart Face Door Unlock System

Submitted in partial fulfillment of award of

BACHELOR OF TECHNOLOGY

Degree
in
(COMPUTER SCIENCE & ENGINEERING)

By

SUMIT KUMAR-(1708210153)
SOHAIL KHAN -(1708210148)
SUMIT DEBNATH -(1708210152)
SUMIT KUMAR -(1708210150)

SUPERVISOR

Ms. Shilpi Rani
Asst. Professor

IN PURSUIT OF EXCELLENCE

Department of Computer Science & Engineering

Moradabad Institute of Technology
Moradabad (U.P.)

Session: (2017-2021)

CERTIFICATE

Certified that the Project Report entitled “ SMART FACIAL DOOR UNLOCK SYSTEM ” submitted by Sohail Khan (1708210147), Sumit Kumar (1708210153), Sumit Debnath (1708210152), Sumit Kumar (1708210150) is their own work and has been carried out under my supervision. It is recommended that the candidates may now be evaluated for their project work by the University.

Shilpi Rani
Assistant Professor

Date:

ABSTRACT

Security is one of the fundamental parts of regular day to day existence. From days of yore, individuals used to give first concern to defend their homes by utilizing diverse entryway locking system. Leave it alone for access into the high-security premises to the regular man house, we generally wish to remain safe and furthermore to shut out the trespassers diminishing concerns. We use from customary lock and key to extremely modern remotely controlled sensor lock. In any case, every one of the locking systems has its negative marks, once in a while it faces the circumstance where we either neglect to bolt the house or additionally in some cases lose or lose the key, if there should be an occurrence of entryways with auto locking system if the keys are lost we have no option aside from cutting the lock . Additionally now and then individuals don't keep the house separate from dread that somebody may burgle. For such reasons, the confidence in the mechanized shrewd entryway lock system is expanding. In the realm of rising innovations; people have turned to biometric opening system however biometric locking system has its weaknesses like palms with slick substances and sweat didn't perceived by the system. The most unsafe factor in biometric system is the surface moved by a few people which can be a hotspot for transmitting savage infections; it is very obvious in the ongoing flare-up of COVID-19 pandemic. To beat every one of these disservices locking system with face acknowledgment is a progressive exertion. So in this exploration, it suggests an entryway lock security system which is fit to open the entryway with a basic snap of our face. This security system has been created by incorporating profound learning and IOT. This security system is created utilizing the Face Recognition, Web Camera, OTP (Using Twilio API), and Solenoid Lock. The unmistakable element of this system is that it utilizes the profound learning face acknowledgment procedure which builds the grouping precision. This Smart Door Unlock System based on Face Recognition to enhance the security. In this system camera sensor or web camera is used to capture the face and image matching algorithm will be used to detect the authenticated faces.

ACKNOWLEDGEMENT

We would like to express my gratitude towards our guide and mentor Associate Prof. **Mrs. Shilpi Rani,** who lit us his knowledge and experience, for all the encouragement and whole-hearted support we acquired during the course of the project and otherwise

Our heartfelt gratitude goes to the HOD of Computer Science and Engineering Department **Dr. Somesh Kumar**, for his enduring support and allowing us to use the institute's infrastructure for the purpose.

We are thankful towards all the PAC Members for their guidance & vision that lead us to successful project implementation. We are also thankful for all the teaching, non-teaching staff, and my classmates for taking interest in discussing my problem and encouraging me. We owe a debt of gratitude to my father and mother for their consistent support, sacrifice, candid views, and meaningful suggestion to me at different stages of this work.

Sohail Khan (1708210147)
Sumit Debnath (1708210152)
Sumit Kumar (1708210153)
Sumit Kumar (1708210150)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
PROJECT OVERVIEW

1.8 INTRODUCTION

Now-a-days with the extreme use of smart devices are used to automate many of the processes. Home automation is one of the aggressively developed technology use by high end society. It's far tough to consider blindly on traditional and simple security features of the device. in conventional gadget many of the doors are having mechanical lock which have been constrained on the number of keys. So, to overcome the aforementioned issues and traditional locking system one has to modify them and make them smart and automated. It works well but when we wish more secured environment and accountability of who locked and unlocked when is the major part was missing in traditional system. This paper proposes Smart Door Unlock System based on Face Detection to enrich the security. Machine learning based approach with Haar Cascade method is proposed in the paper. In this system camera sensor or web camera will be used to capture the face and image matching algorithm will be used to detect the authenticated faces. Only the person whose face is matched can be able to unlock the door. So, limitation of dealing with keys will be resolved. This system will now not best beautify the safety however additionally make the device keyless. Many promising digital based automated solutions came in market whose detailed analysis is given in literature survey, a few are thumb based, Iris based and Face Based. Many people tried to develop the automation on door based on smart cands, thumb based, iris based but very few of them are prominent for face based solution .

This system is so promising but has its own pros and cons. Certain challenges are also faced when we use face detection such as lightening, varying brightness. The main advantage of this system is acquiring the door using face detection approach and entire face is recognized. Face recognition technique involves attribute extraction from facial image with help of smart door model an intense innocence is expected in security industry and to make daily objects synergistic.

1.9 LITERATURE SURVEY

Today people are facing more problems about security in all over world, nowadays security is the most essential issue everywhere in the world; so security of everything gains higher and higher importance in recent years. Here in this paper, trying to reproduce the comprehensive literature study related to the various door locks and gate security systems that are necessary in the fields such as home, industries and vehicle security where possibilities of incursion are increasing day by day. In past days, the research is gone on various door lock security systems like traditional security systems which provide indications using alarm. Due to the advancement in recent techniques, some door lock security systems are based on microcontroller, GSM, GPS, many sensors, software like MATLAB, PROTEUS, biometrics like face recognition, Iris scanner, RFID, Smart Card and password etc. Each system has their own advantages and disadvantages. In most of systems, SMS technique is used for communication so the system will become cost effective, more reliable and it will take less time to deliver message.

Door lock security systems are classified based on technology used 1) GSM based, 2) smart card based, 3) Password based, 4) Biometric based, 5) RFID based, 6) Door phone based, 7) Bluetooth based, 8) Social networking sites based, 9) OTP based, 10) Motion detector based, 11) VB based, 12) Combined system.

1.9.1 Password Based System

The programmable electronic code lock device is programmed in such a way that it will operates only with the correct entry of predefined digits. It is also called an integrated combinational type lock. The programmable code lock is shown in Fig 1 as below.
Electronics safe is its example. Based on the programmable electronic code lock, the reprogrammable digital door locks [2] were invented in that the password can change any time as it stored in PROM. For operating the device, GSM/CDMA module can be used. When any person calls up from his phone, the call will be received by the system. And the door will opens only if the call is from specified user.

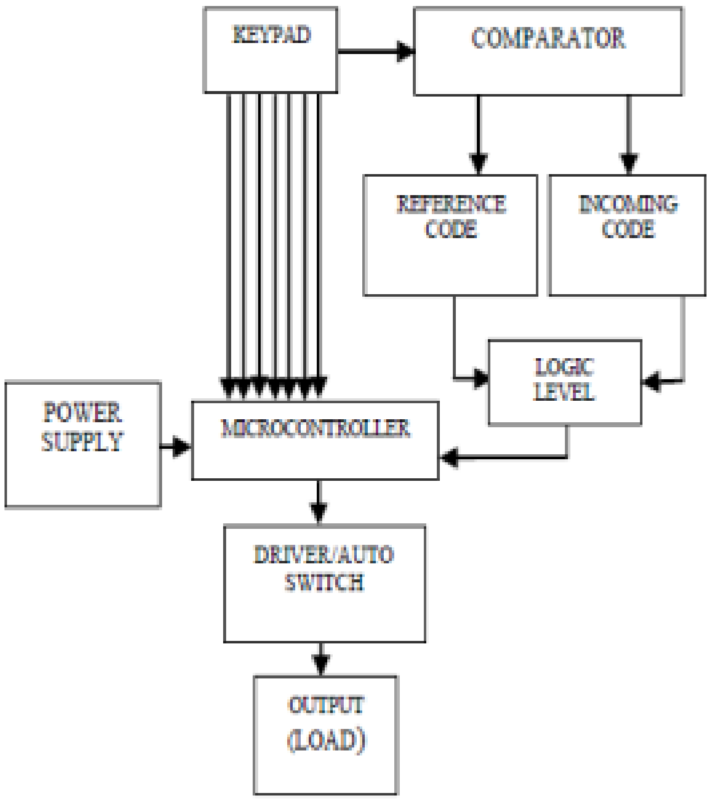


Fig 1: Programmable Electronic Code Lock

A cell phone controlled password protected door lock system is as shown in Fig 1 which was proposed to open the door with the help of cell phone device by entering a specific code. The user can make a call to a system's number. This call is responsible for opening or closing of the entry with the use of correct password. In latest password based system, a more advanced system develops which communicates the owner of the office or house, when any unauthorized person tries to open the code, by giving correct code as well. While closing the door of office/home, the owner has to press the 0 key available on the hex keypad and free the system. The system developed by Annie P. Oommen et. al. allows for changing the password. To open the lock, the entered password must matches with the changed one. In some systems the security dial-up enables through the GSM modem when the unauthorized person enters an invalid password then the controller informs to the owner through GSM modem. Latest security system is designed where the locking security system can be enhanced with the help of GSM and RF wireless technology by using a 4 digit password which provides the authentication.

1.9.2 Biometric Based System

The palmtop recognition is the next step for fingerprint recognition. It operates on the image of palmtop. Firstly system takes an image of the palmtop then it works on that image by partitioning it and process is required. At the end, verify the right person. Hence, it reduces the chances of error in other human recognition methods and clarifies the problems which were faced in the fingerprint recognition. The biometric technique is very useful in bank lockers. Except fingerprint recognition the vein detector and iris scanner gives best and accurate result so, in the bank security system, microcontroller continuously observes the Vein Detector and Iris Scanner through keypad authenticated codes. During night the wireless motion detector will be active, if any variation occurs in its output, it will be sensed by the controller and alert sounds will be given by it. Recently, the fast based principal component analysis approach is proposed in which the modification of principal component analysis approach for the face recognition and face detection process is done. The image is captured by the web camera and it gets matched with the image stored in the database. New advanced door lock security systems are available based on the pattern of the human iris for providing a high level of security. And to make the system more efficient n reliable the duplication is done in MATLAB.

1.2.3 GSM Based Systems

In many door lock security systems, GSM is used for communication purpose. The purpose of a work cultivated by utilization of a circuits like a GSM module which gets activated by a controller for sending SMS in emergency to proprietor and for sending corresponding services of security at the time of break in. For detecting obstacles, the system requires various sensors. It gathers data from the sensors and settles on a choice. With the help of GSM module, sends SMS to a respective number. A recently created model for security of door easily controlled like remote control operations by a GSM hand set acts as the transmitter and the other GSM phone set with the DTMF associated with the motor attached to door with the use of DTMF decoder, a stepper motor and microcontroller unit. Nowadays people want to be secure though they are away from home so, the work proposed by Jayashri Bangali et. al. . When the owner is not at his home, security of home and important things is the big issue in front of all. Two frameworks were created which depends on GSM based technology. For detection of the gate-crashes, it takes place by capturing image through web camera. When peoples are not at their homes, the system sends notification in terms of SMS to the crisis number. A novel administrator based system can login without any stretch to the system and can see guests record and listen their recorded messages and also automatically lock the door using mobile communication technology.

1.3 PROBLEM STATEMENT

Security and safety are becoming more and more popular day by day. Hence, it is getting improved and used for the ease in our life. Nowadays, technology has become an integrated part of people's lives therefore the security of one's home, office or organization must not be left behind. Our goal is to design a system with minimum human intervention. Smart Receptionist with smart lock system is mainly designed and developed for security and safety purposes. This smart security system is used to see a visitor when the main door of the office or organization is closed. The purpose of this system is to control the door lock using IOT and ML. In this system, whenever a person rings the bell, image of the person is captured by the camera. If person's image is present in database, he/she will be allowed in else if image is not present in the database then the smart lock system will send the image to system. Once the image is uploaded on the system, option will be available with user of either locking or unlocking the door. If the user clicks on unlock button the door will open and person standing on the door will be allowed in and if the user clicks on lock button the door will remain locked and person will not be allowed in. door and the system will display as "Access Denied".

1.4 METHODOLOGY

The aim is to provide a high security face detection system and send an alert to the authorized person via Buzzer module. The flow is as follows:

- Interfacing of input unit to capture live Face image.
- Create a database of authorized house members and relatives.
- Capture the face at the doorstep and compare with data base image.
- System send an OTP to the person in authorization.
- Interface relay to the lock system.

1.5 FACE RECOGNITION SYSTEM

A pattern recognition task performed exclusively on faces is termed as face recognition. It can be described as classifying a face either known or unknown, after matching it with stored known individuals as a database. It is also advantageous to have a system that has the capability of learning to identify unknown faces. The There are five main functional blocks, whose responsibilities are as below.

A. **The acquisition module** This is the entry point of the face recognition process. The user gives the face image as the input to face recognition system in this module.

B. **The pre-processing module** In this module the images are normalized to improve the recognition of the system. The pre-processing steps implemented are as follows: • Image size normalization • Background removal • Translation and rotational normalizations • Illumination normalization

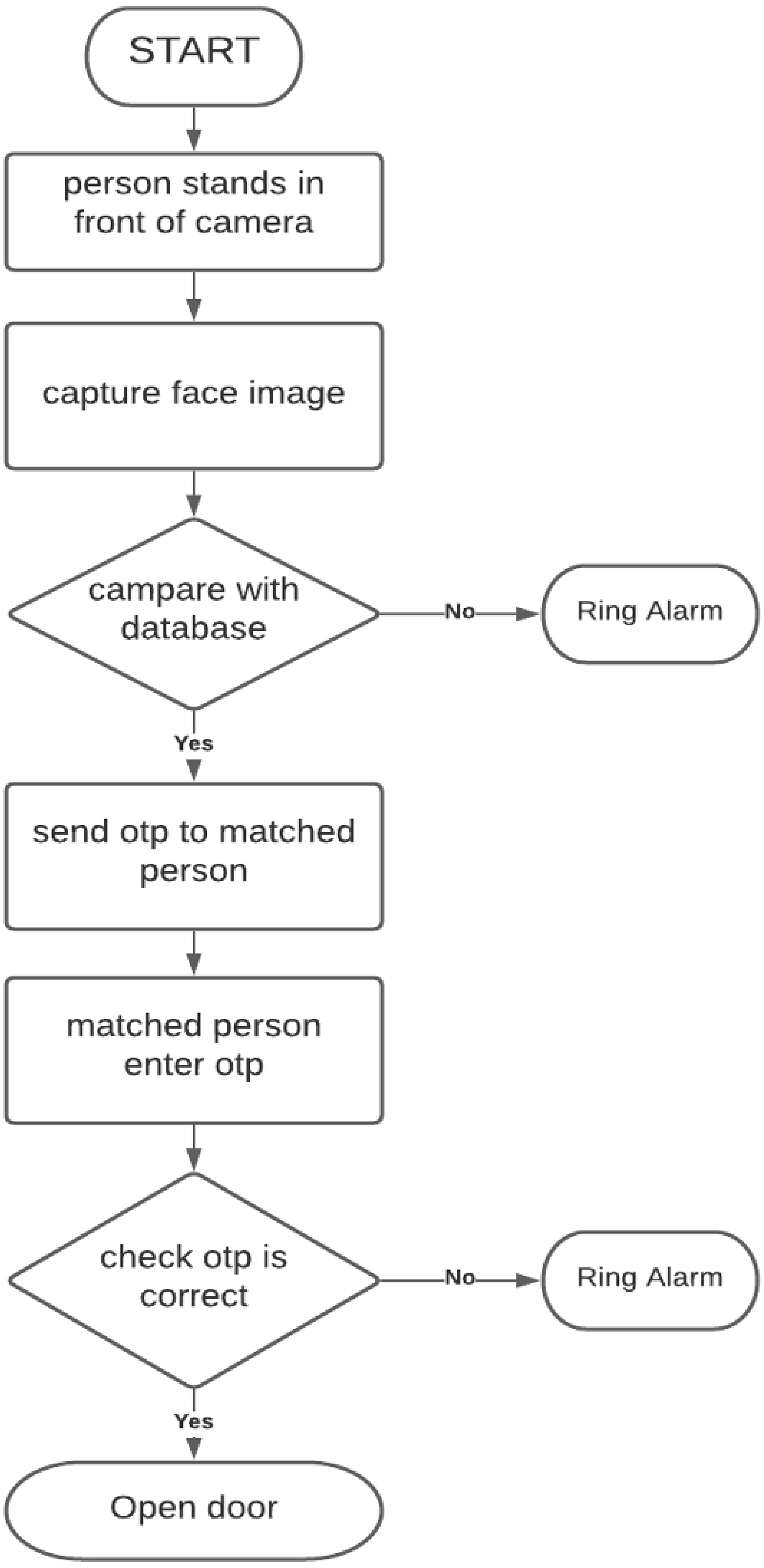
C. **The feature extraction module** After the preprocessing the normalized face image is given as input to the feature extraction module to find the key features that will be used for classification. The module composes a feature vector that is well enough to represent the face image.

D. **The classification module** With the help of a pattern classifier, the extracted features of face image are compared with the ones stored in the face database. The It is used to match the test face image is then classified as either known or unknown. E. Face database image with the train images stored in a database. If the face is recognized as "unknown", face images can then be added to the database for further comparisons.

1.6 ONE TIME PASSWORD (OTP)

This is the one time password which add the extra security to our proposed security system. When the system recognize the face then it will send a OTP to the person registered phone number. And that person have only 3 chances if he/she failed to enter the OTP 3 times then the system will block him/her entry and alert about this failure.

1.7 FLOW CHART



CHAPTER 2
HARDWARE AND SOFTWARE REQUIREMENTS

2.1 SOFTWARE REQUIREMENTS

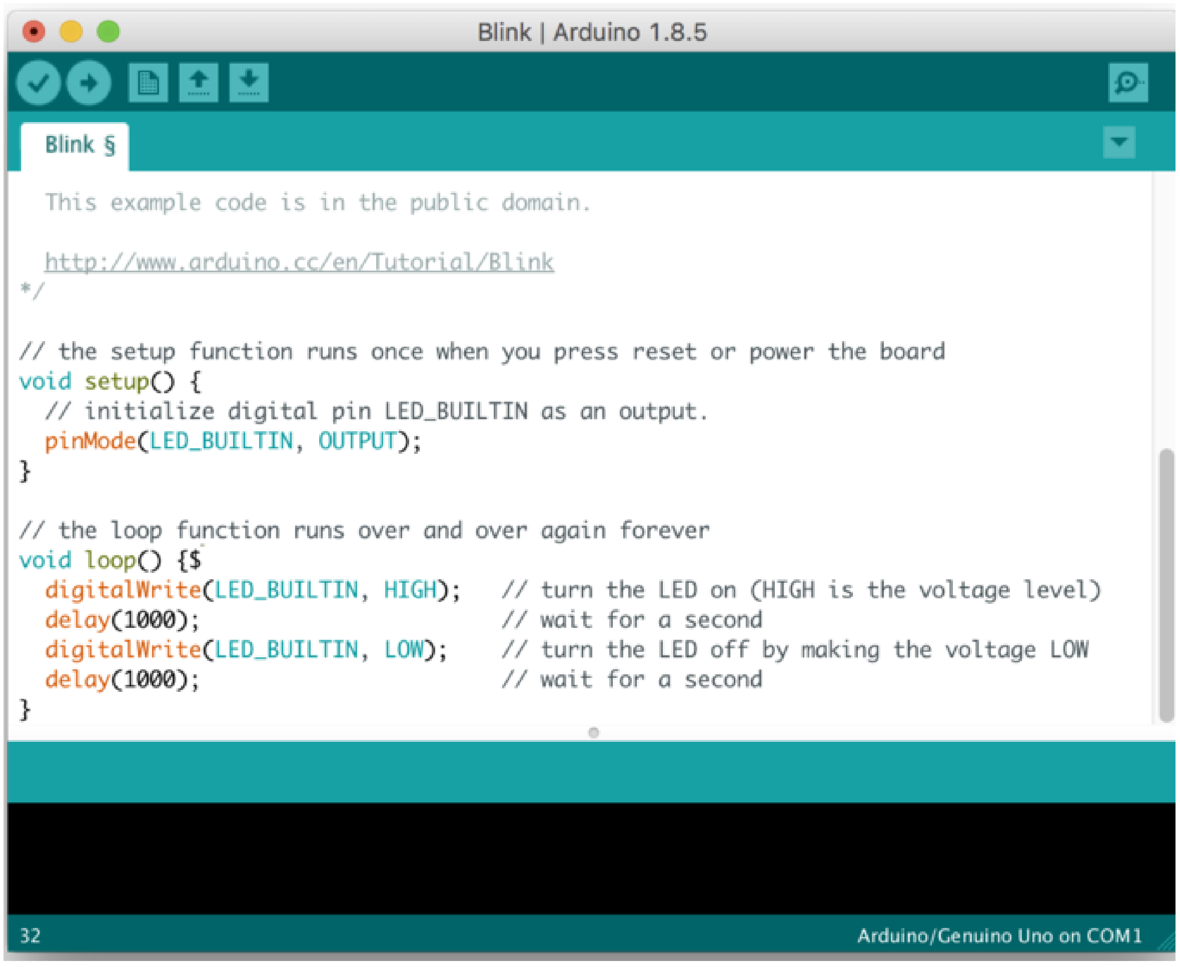
In our project, software requirements are very less due to its more dependency on hardware but we need some software and web services that are required to operate this whole system and for additional benefits to the user. Some Software dependences are listed below:

2.1.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two 13 basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution.

The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards. With the rising popularity of Arduino as a software platform, other vendors started to implement custom open source compilers & tools (cores) that can build and upload sketches to other MCUs that are not supported by Arduino's official line of MCUs.



```
  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {$
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Fig 2.1: ARDUINO IDE

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

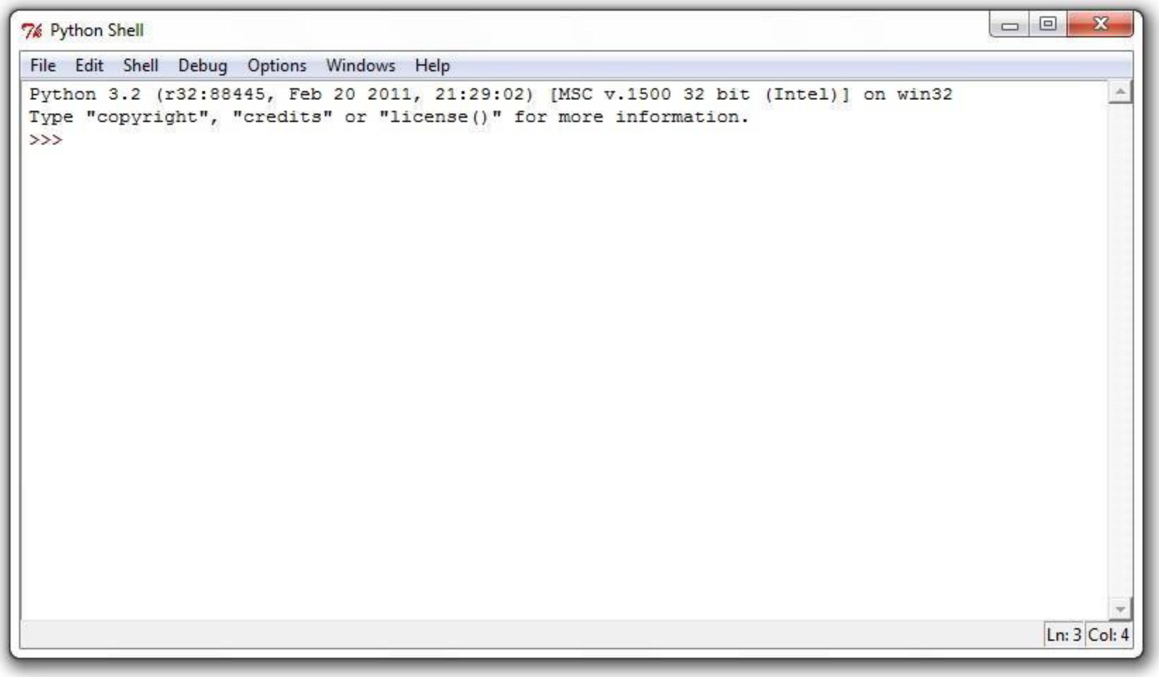
2.1.2 Python IDLE

Python is an interpreted high-level general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs as well as its object- oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

IDLE is Python’s Integrated Development and Learning Environment. IDLE has the following features:

- coded in 100% pure Python, using the tkinter GUI toolkit
- cross-platform: works mostly the same on Windows, Unix, and macOS
- Python shell window (interactive interpreter) with colorizing of code input, output, and error messages
- multi-window text editor with multiple undo, Python colorizing, smart indent, call tips, auto completion, and other features
- search within any window, replace within editor windows, and search through multiple files (grep)
- debugger with persistent breakpoints, stepping, and viewing of global and local namespaces
- configuration, browsers, and other dialogs



```
Python 3.2 (r32:88445, Feb 20 2011, 21:29:02) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

Fig 2.2 PYTHON IDE

2.1.3 OpenCV

OpenCV is a Python library that was designed to solve computer vision problems. OpenCV was originally developed in 1999 by Intel, but later, it was supported by Willow Garage. It supports a wide variety of programming languages such as C++, Python, Java, etc. Support for multiple platforms including Windows, Linux, and MacOS.

OpenCV Python is nothing but a wrapper class for the original C++ library to be used with Python. Using this, all of the OpenCV array structures gets converted to/from NumPy arrays. This makes it easier to integrate it with other libraries that use NumPy. For example, libraries such as SciPy and Matplotlib.

Gary Bradsky started OpenCV at Intel in 1999. While it supports a gamut of languages like C++, Python, and more, and OpenCV-Python is an API for OpenCV to unleash the power of Python and the OpenCV C++ API at once.

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

2.1.4 PYTHON GUI TKINTER

Tkinter is the most commonly used library for developing GUI (Graphical User Interface) in Python. It is a standard Python interface to the Tk GUI toolkit shipped with Python. As Tk and Tkinter are available on most of the Unix platforms as well as on the Windows system, developing GUI applications with Tkinter becomes the fastest and easiest.

You don’t need to worry about the installation of the Tkinter module separately as it comes with Python already. It gives an object-oriented interface to the Tk GUI toolkit. Graphical User Interface (GUI) is a form of user interface which allows users to interact with computers through visual indicators using items such as icons, menus, windows, etc. It has advantages over the Command Line Interface (CLI) where users interact with computers by writing commands using keyboard only and whose usage is more difficult than GUI.

Python has a lot of GUI frameworks, but Tkinter is the only framework that’s built into the Python standard library. Tkinter has several strengths. It’s cross-platform, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they’re run.
Although Tkinter is considered the de-facto Python GUI framework, it’s not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you’re looking for.

However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern sheen is unnecessary, and the top priority is to build something that’s functional and cross-platform quickly.

2.2 HARDWARE REQUIREMENTS

As our project is based on IOT (Internet of Things) and (Face Recognition) Machine Learning (ML) we have used lot of hardware components in the form of sensors and microcontrollers. Due to large dependency on IoT Modules and microcontroller this project having a hardware and response system are highly depends upon the hardware components. Hardware components are explained in further sections.

2.2.1 Microcontroller

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip.

Sometimes referred to as an embedded controller or microcontroller unit (MCU), microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances, among other devices. They are essentially simple miniature personal computers (PCs) designed to control small features of a larger component, without a complex front-end operating system (OS).

A microcontroller is embedded inside of a system to control a singular function in a device. It does this by interpreting data it receives from its I/O peripherals using its central processor. The temporary information that the microcontroller receives is stored in its data memory, where the processor accesses it and uses instructions stored in its program memory to decipher and apply the incoming data. It then uses its I/O peripherals to communicate and enact the appropriate action.

Microcontrollers are used in a wide array of systems and devices. Devices often utilize multiple microcontrollers that work together within the device to handle their respective tasks.

For example, a bike might have many microcontrollers that control various individual systems within, such as the anti-lock braking system, traction control, fuel injection or suspension control. All the microcontrollers communicate with each other to inform the correct actions. Some might communicate with a more complex central computer within the bike, and others might only communicate with other microcontrollers. They send and receive data using their I/O peripherals and process that data to perform their designated tasks. The core elements of a microcontroller are:

- The processor (CPU) -- A processor can be thought of as the brain of the device. It processes and responds to various instructions that direct the microcontroller's function. This involves performing basic arithmetic, logic and I/O operations. It also performs data transfer operations, which communicate commands to other components in the larger embedded system.

- Memory -- A microcontroller's memory is used to store the data that the processor receives and uses to respond to instructions that it's been programmed to carry out. A microcontroller has two main memory types:

    1. Program memory, which stores long-term information about the instructions that the CPU carries out. Program memory is non-volatile memory, meaning it holds information over time without needing a power source.

    2. Data memory, which is required for temporary data storage while the instructions are being executed. Data memory is volatile, meaning the data it holds is temporary and is only maintained if the device is connected to a power source.

- I/O peripherals -- The input and output devices are the interface for the processor to the outside world. The input ports receive information and send it to the processor in the form of binary data. The processor receives that data and sends the necessary instructions to output devices that execute tasks external to the microcontroller.

- While the processor, memory and I/O peripherals are the defining elements of the microprocessor, there are other elements that are frequently included. The term I/O peripherals itself simply refers to supporting components that interface with the memory and processor. There are many supporting components that can be classified as peripherals. Having some manifestation of an I/O peripheral is elemental to a microprocessor, because they are the mechanism through which the processor is applied.

Other supporting elements of a microcontroller include:

- Analog to Digital Converter (ADC) -- An ADC is a circuit that converts analog signals to digital signals. It allows the processor at the center of the microcontroller to interface with external analog devices, such as sensors.

- Digital to Analog Converter (DAC) -- A DAC performs the inverse function of an ADC and allows the processor at the center of the microcontroller to communicate its outgoing signals to external analog components.

- System bus -- The system bus is the connective wire that links all components of the microcontroller together.

- Serial port -- The serial port is one example of an I/O port that allows the microcontroller to connect to external components. It has a similar function to a USB or a parallel port but differs in the way it exchanges bits.

2.2.1.1 Node MCU

NodeMCU is a low-cost open source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module.Later, support for the ESP32 32-bit MCU was added. NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit). The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source. The firmware uses the Lua scripting language.

The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

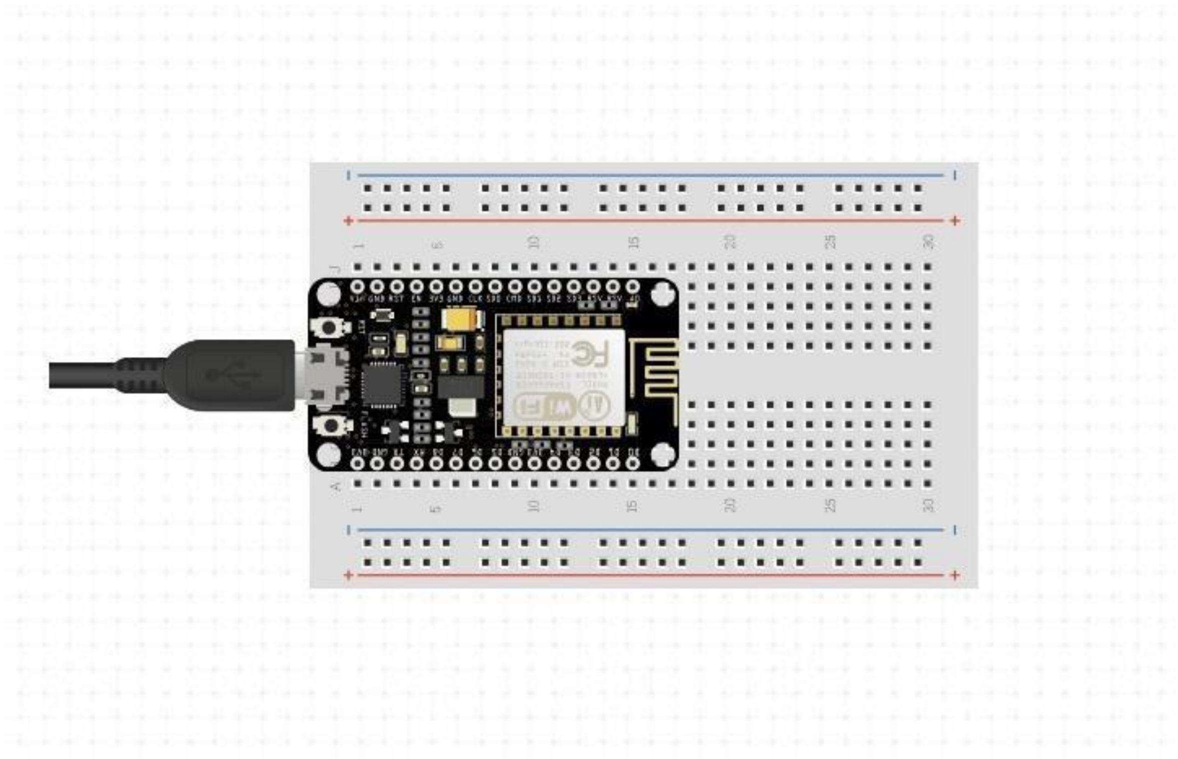
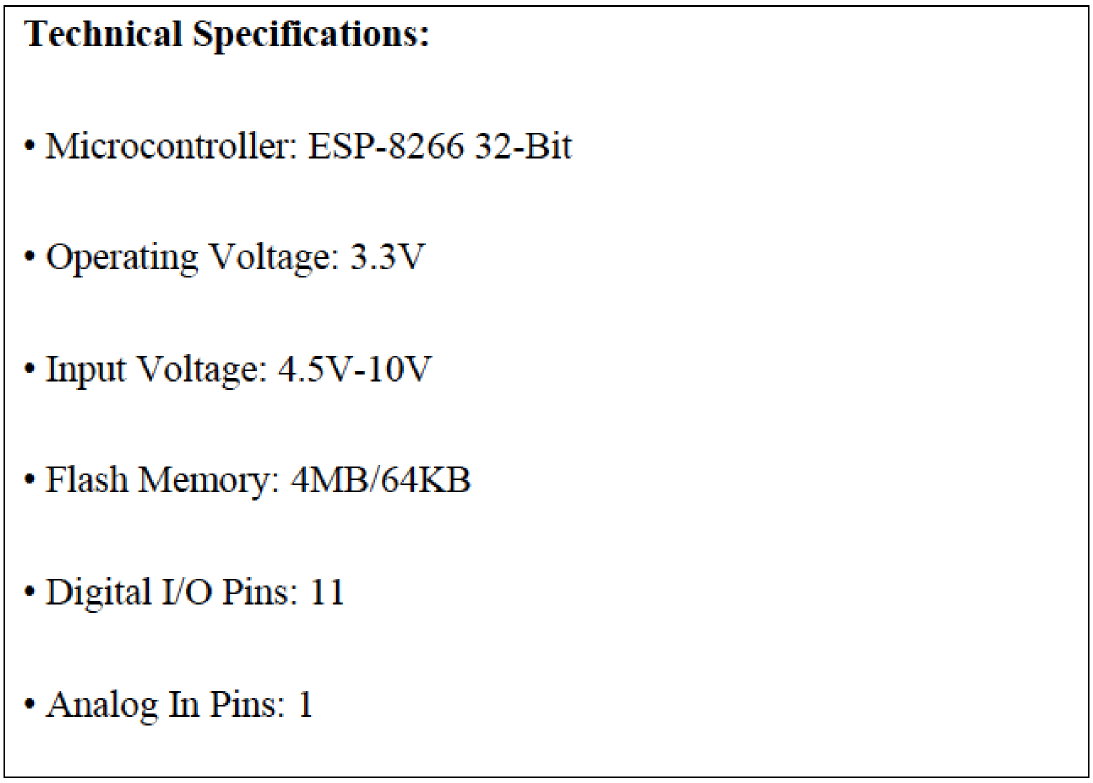


Fig 2.3: NODE MCU

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface- mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially was based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects).


Technical Specifications:

• Microcontroller: ESP-8266 32-Bit

• Operating Voltage: 3.3V

• Input Voltage: 4.5V-10V

• Flash Memory: 4MB/64KB

• Digital I/O Pins: 11

• Analog In Pins: 1

Fig 2.4 TECHNICAL SPECIFIACTION

Pins

- **Power**
  - Micro-USB: NodeMCU can be powered through the USB port
  - 3.3V: Regulated 3.3V can be supplied to this pin to power the board

    - GND: Ground pins o Vin: External Power Supply • Control Pins o EN, RST: The pin and the button resets the microcontroller

- **Analog Pin**

    - A0: Used to measure analog voltage in the range of 0-3.3V.

- **GPIO Pins**
    - GPIO1 to GPIO16: NodeMCU has 16 general purpose input-output pins on its board.

- **SPI Pins**

    - SD1, CMD, SD0, CLK: NodeMCU has four pins available for SPI communication.

- **UART Pins:**

    - TXD0, RXD0, TXD2, RXD2: NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program.

- **I2C Pins**

    - NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C.

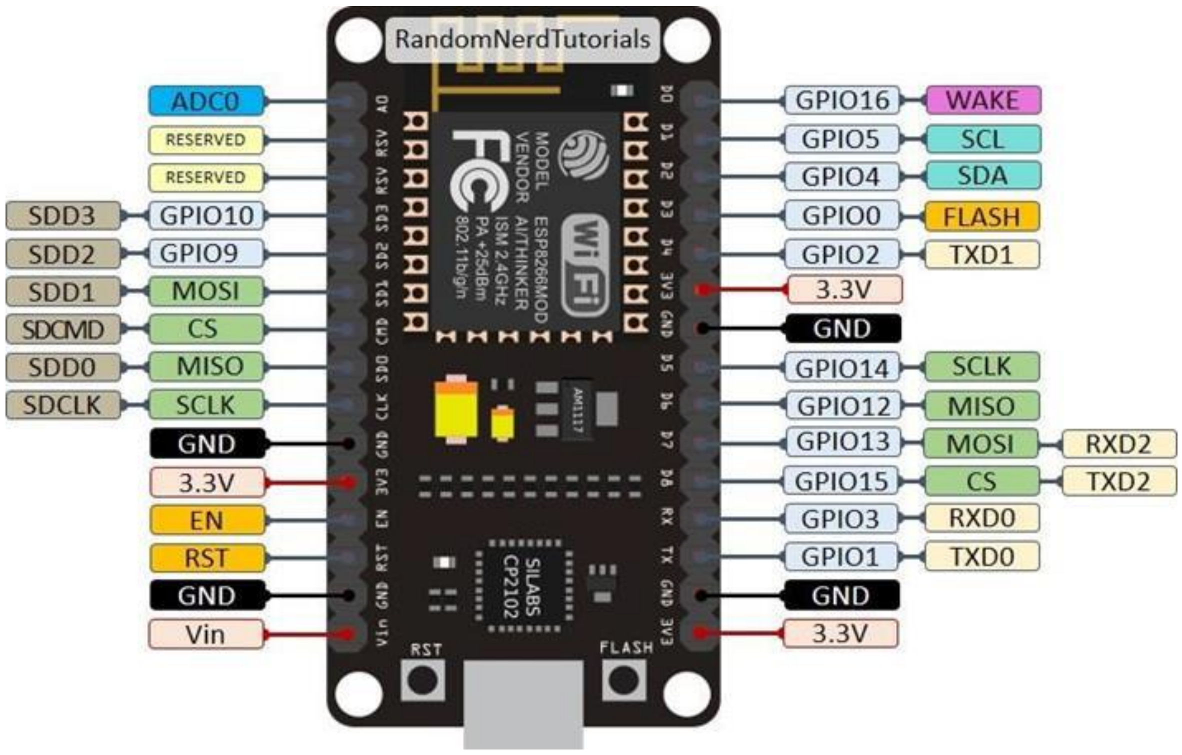


Fig 2.5: NODEMCU PIN DIAGRAM

2.2.2 Solenoid Lock

The solenoid lock denotes a latch for electrical locking and unlocking. It is available in unlocking in the power-on mode type, and locking and keeping in the power-on mode type, which can be used selectively for situations. The power-on unlocking type enables unlocking only while the solenoid is powered on. A door with this type is locked and not opened in case of power failure or wire disconnection, ensuring excellent safety. This type is used mainly for places requiring crime prevention. The power-on locking type can lock a door while the solenoid is powered on. If the power is disconnected, the door is unlocked.

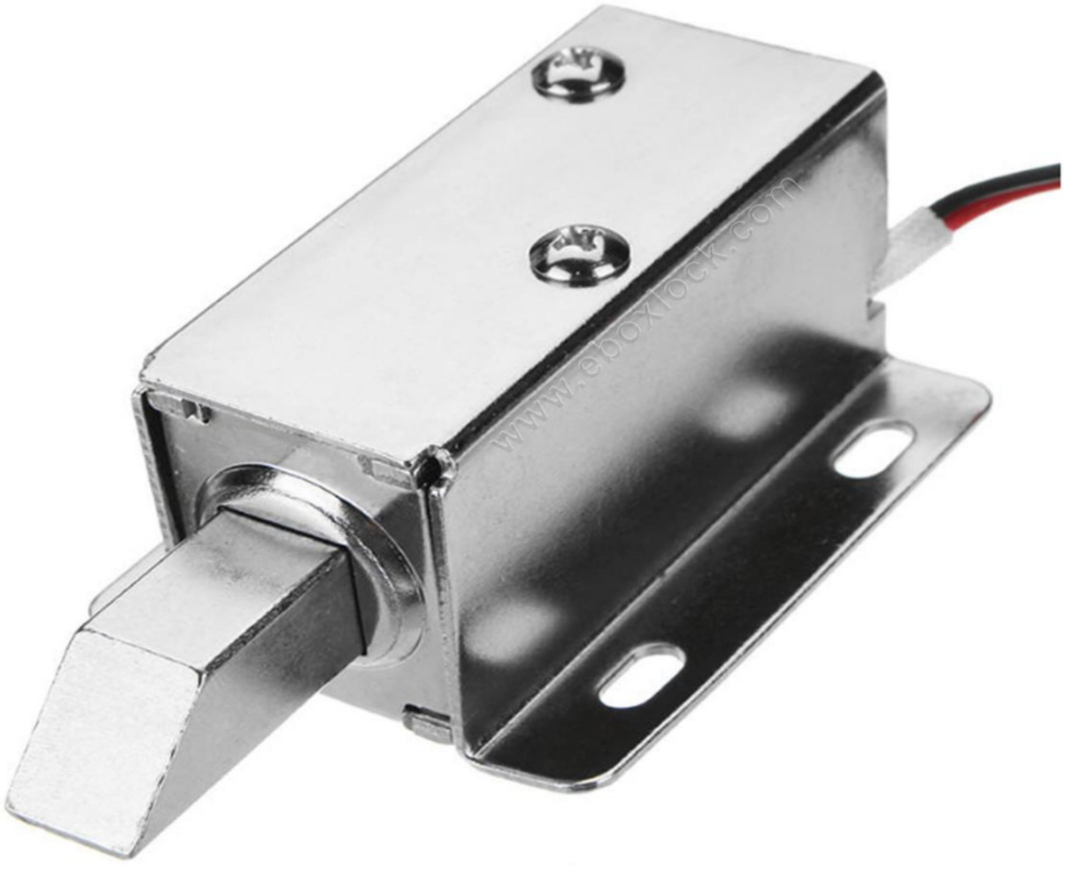


Fig 2.6: SOLENOID LOCK

A solenoid lock works on the electronic-mechanical locking mechanism. This type of lock has a slug with a slanted cut and a good mounting bracket. When the power is applied, DC creates a magnetic field that moves the slug inside and keeps the door in the unlocked position. The slug will retain its position until the power is removed. When the power is disconnected, the slug moves outside and locks the door. It doesn't use any power in a locked state. To drive the solenoid lock, you would need a power source that can give 12V @ 500mA.

This type unlocks the door in case of wire disconnection due to a fire or accident, and it is used for emergency exits through which fire-fighting activity or evacuation should preferentially be made rather than safety for crime prevention. The keeping type performs two operations, locking and unlocking by applying a positive or negative pulse voltage to the solenoid, and keeps the no-power state in each position. This type features energy saving because it is unnecessary to always power the solenoid on. For the continuous rating and the intermittent rating, the continuous rating is designed to be able to feed a rated voltage power continuously for hours without exceeding a specified temperature rise limit, and the intermittent rating is designed to be able to feed a specified voltage only for a specified time duration without exceeding a specified temperature rise limit.

2.2.6 Relay Module

A power relay module is an electrical switch that is operated by an electromagnet. The electromagnet is activated by a separate low-power signal from a micro controller. When activated, the electromagnet pulls to either open or close an electrical circuit. A simple relay consists of wire coil wrapped around a soft iron core, or solenoid, an iron yoke that delivers a low reluctance path for magnetic flux, a movable iron armature and one or more sets of contacts. The movable armature is hinged to the yoke and linked to one or more set of the moving contacts. Held in place by a spring, the armature leaves a gap in the magnetic circuit when the relay is de-energized. While in this position, one of the two sets of contacts is closed while the other set remains open. When electrical current is passed through a coil, it generates a magnetic field that in turn activates the armature. This movement of the movable contacts makes or breaks a connection with the fixed contact. When the relay is de-energized, the sets of contacts that were closed, open and breaks the connection and vice versa if the contacts were open.

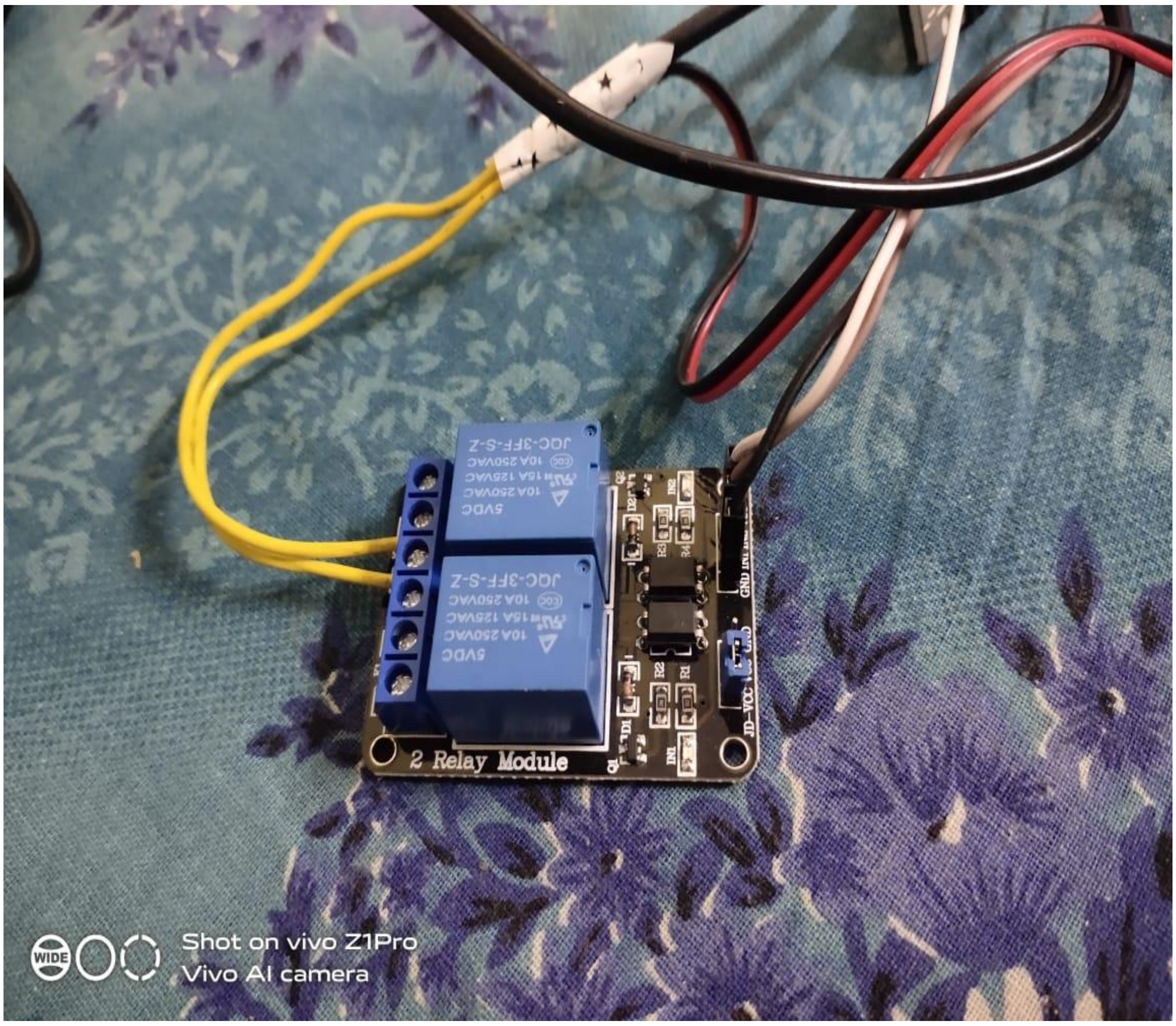


Fig 2.7: RELAY MODULE

When switching off the current to the coil, the armature is returned, by force, to its relaxed position. This force is usually provided by a spring, but gravity can also be used in certain applications. Most power relays are manufactured to operate in a quick manner. For distribution of power in high current applications, GEP Power Products is the industry leader in high power relay module design and manufacturing. Rated up to 70 amps, GEP’s power relay modules are designed for seamless integration in high power distribution applications. The convenient integral mounting brackets provide easy installation and accessibility. With endless options such as terminal position assurance available for wire retention, GEP Power Products’ power distribution solutions and off-road industry knowledge are second to none.

2.2.7 Buzzer

A buzzer is a mechanical, electromechanical, magnetic, electromagnetic, electro-acoustic or piezoelectric audio signalling device. A piezo electric buzzer can be driven by an oscillating electronic circuit or other audio signal source. A click, beep or ring can indicate that a button has been pressed.

Fig 2.8: Buzzer

2.2.7.1 Types of Buzzer

There are several different kinds of buzzers. At Future Electronics, we stock many of the most common types categorized by Type, Sound Level, Frequency, Rated Voltage, Dimension, and Packaging Type. The parametric filters on our website can help refine your search results depending on the required specifications. The most common sizes for Sound Level are 80 dB, 85 dB, 90 dB and 95 db. We also carry buzzers with Sound Level up to 105 db. There are several types available including Electro-Acoustic, Electromagnetic, Electromechanics, Magnetic and Piezo, among others.

2.2.8 Jumpers Wires

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed. Fairly simple. In fact, it doesn't get much more basic than jumper wires.

Fig 2.9 JUMPERS WIRE

CHAPTER 3
FACE DETECTION

The problem of face recognition is all about face detection. This is a fact that seems quite bizarre to new researchers in this area. However, before face recognition is possible, one must be able to reliably find a face and its landmarks. This is essentially a segmentation problem and in practical systems, most of the effort goes into solving this task. In fact the actual recognition based on features extracted from these facial landmarks is only a minor last step.

There are two types of face detection problems:

1) Face detection in images and
2) Real-time face detection

3.1 FACE DETECTION IN IMAGES

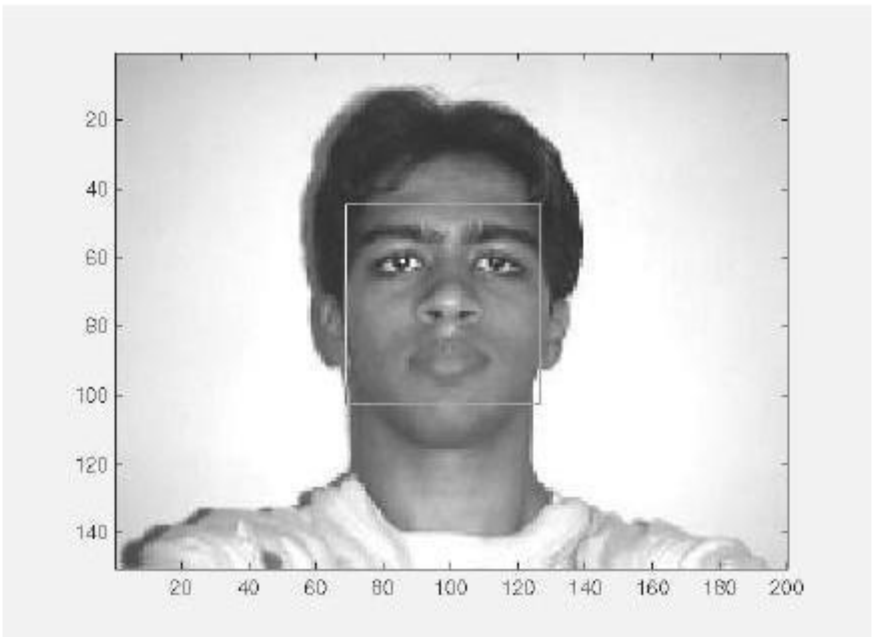


Figure 3.1 A successful face detection in an image with a frontal view of a human face.

Most face detection systems attempt to extract a fraction of the whole face, thereby eliminating most of the background and other areas of an individual's head such as hair that are not necessary for the face recognition task. With static images, this is often done by running a across the image. The face detection system then judges if a face is present inside the window (Brunelli and Poggio, 1993). Unfortunately, with static images there is a very large search space of possible locations of a face in an image

Most face detection systems use an example based learning approach to decide whether or not a face is present in the *window* at that given instant (Sung and Poggio,1994 and Sung,1995). A neural network or some other classifier is trained using supervised learning with 'face' and 'nonface' examples, thereby enabling it to classify an image (*window* in face detection system) as a 'face' or 'non-face'.. Unfortunately, while it is relatively easy to find face examples, how would one find a representative sample of images which represent non-faces (Rowley et al., 1996)? Therefore, face detection systems using example based learning need thousands of 'face' and 'nonface' images for effective training. Rowley, Baluja, and Kanade (Rowley et al.,1996) used 1025 face images and 8000 non-face images (generated from 146,212,178 sub-images) for their training set!

There is another technique for determining whether there is a face inside the face detection system's *window* - using Template Matching. The difference between a fixed target pattern (face) and the window is computed and thresholded. If the window contains a pattern which is close to the target pattern(face) then the window is judged as containing a face. An implementation of template matching called Correlation Templates uses a whole bank of fixed sized templates to detect facial features in an image (Bichsel, 1991 & Brunelli and Poggio, 1993). By using several templates of different (fixed) sizes, faces of different scales (sizes) are detected. The other implementation of template matching is using a deformable template (Yuille, 1992). Instead of using several fixed size templates, we use a deformable template (which is non-rigid) and there by change the size of the template hoping to detect a face in an image.

A face detection scheme that is related to template matching is image invariants. Here the fact that the local ordinal structure of brightness distribution of a face remains largely unchanged under different illumination conditions (Sinha, 1994) is used to construct a spatial template of the face which closely corresponds to facial features. In other words, the average grey-scale intensities in human faces are used as a basis for face detection. For example, almost always an individual’s eye region is darker than his forehead or nose. Therefore an image will match the template if it satisfies the 'darker than' and 'brighter than' relationships (Sung and Poggio, 1994).

3.2 REAL-TIME FACE DETECTION

Real-time face detection involves detection of a face from a series of frames from a video capturing device. While the hardware requirements for such a system are far more stringent, from a computer vision stand point, real-time face detection is actually a far simpler process than detecting a face in a static image. This is because unlike most of our surrounding environment, people are continually moving. We walk around, blink, fidget, wave our hands about, etc.

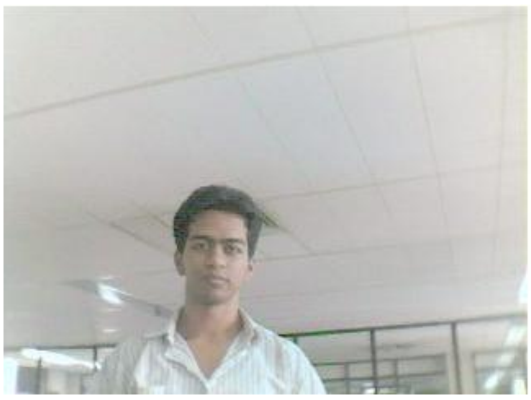


Figure 3.2.1: Frame 1 from camera

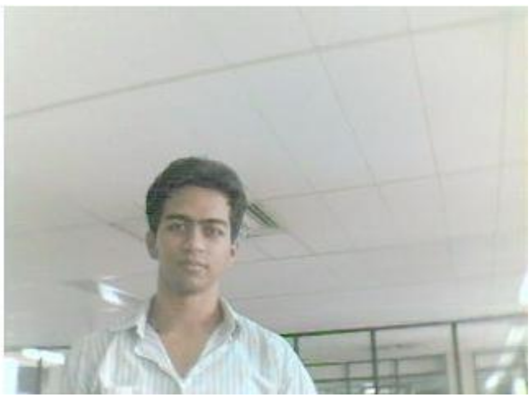


Figure 3.2.2: Frame 2 from camera

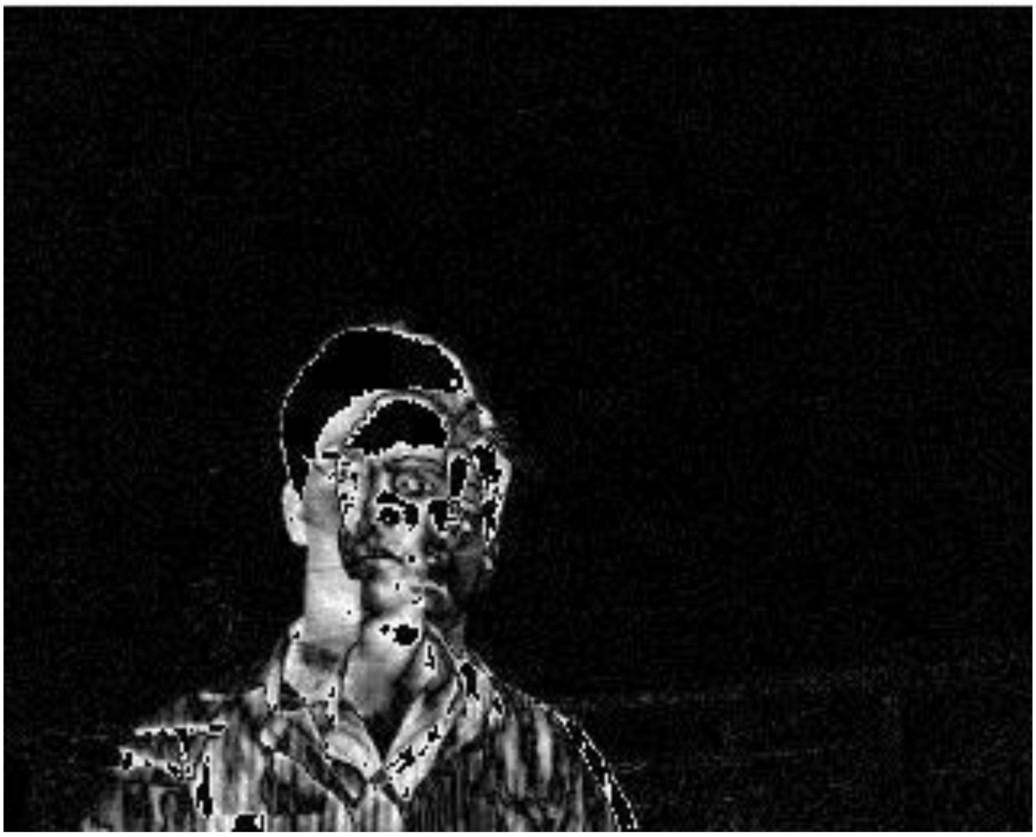


Figure 3.2.3: Spatio-Temporally filtered image

Since in real-time face detection, the system is presented with a series of frames in which to detect a face, by using spatio-temperal filtering (finding the difference between subsequent frames), the area of the frame that has changed can be identified and the individual detected (Wang and Adelson, 1994 and Adelson and Bergen 1986).Further more as seen in Figure exact face locations can be easily identified by using a few simple rules, such as,

- the head is the small blob above a larger blob -the body
- head motion must be reasonably slow and contiguous -heads won't jump around erratically (Turk and Pentland 1991a, 1991b).

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

3.3 FACE DETECTION PROCESS

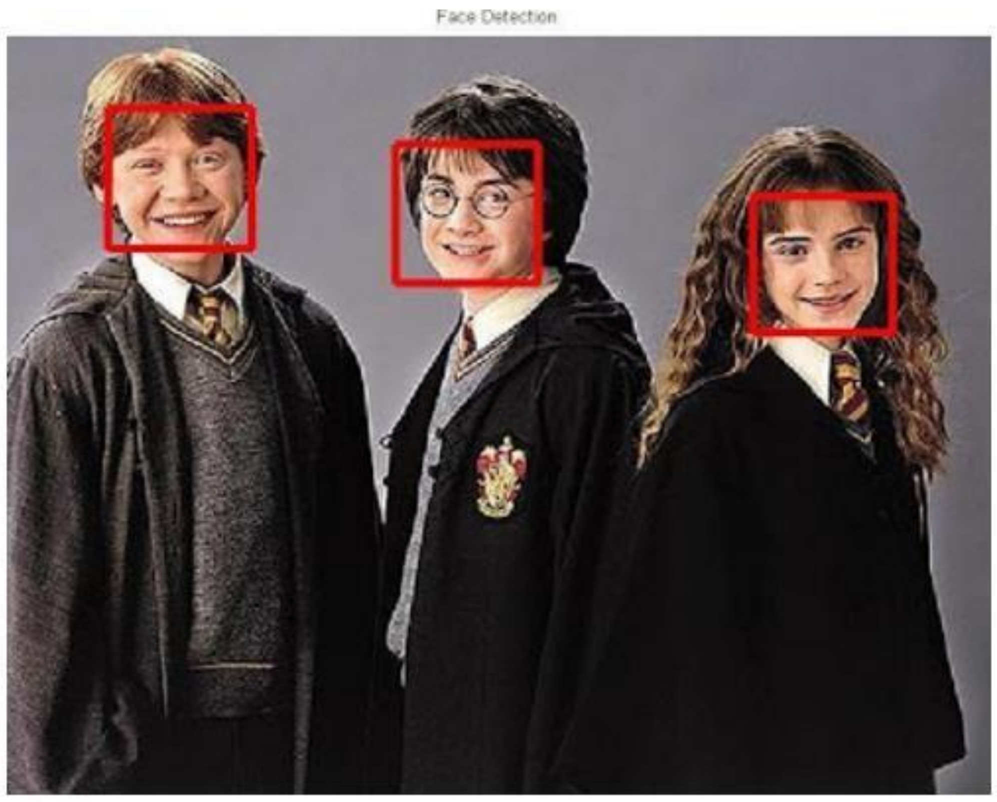
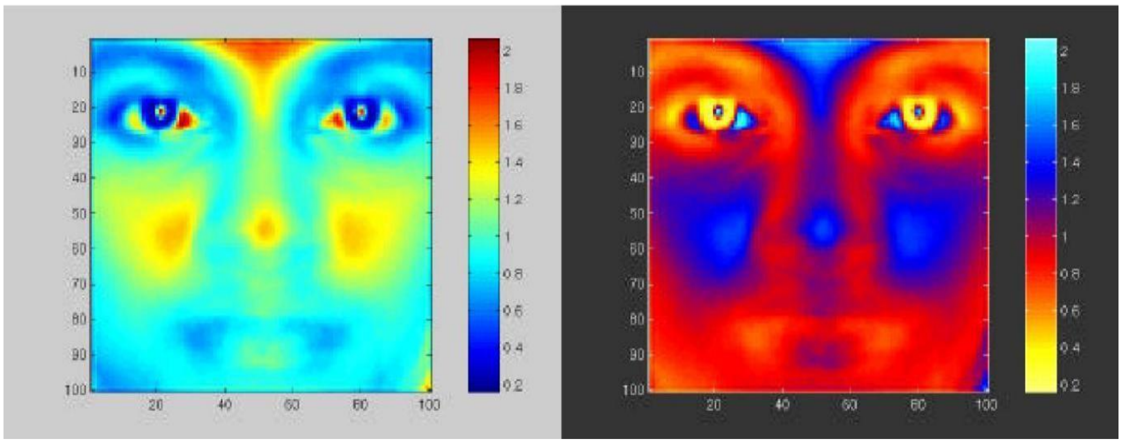


Fig 3.3 Face detection

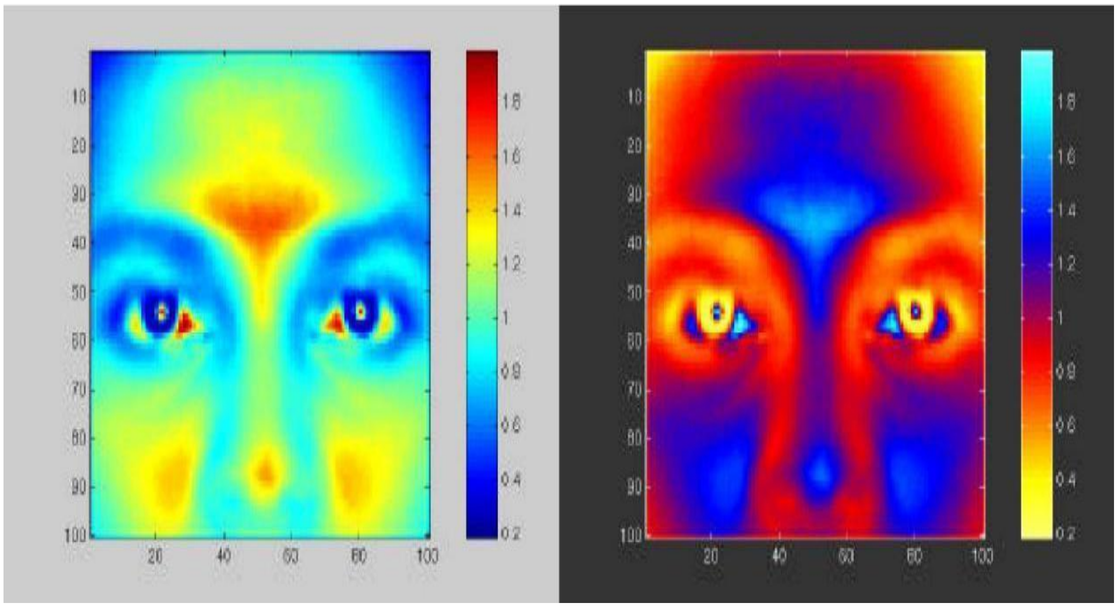
It is process of identifying different parts of human faces like eyes, nose, mouth, etc… this process can be achieved by using MATLAB codeIn this project the author will attempt to detect faces in still images by using image invariants. To do this it would be useful to study the greyscale intensity distribution of an average human face. The following 'average human face' was constructed from a sample of 30 frontal view human faces, of which 12 were from females and 18 from males. A suitably scaled colormap has been used to highlight grey-scale intensity differences.



scaled colormap scaled colormap (negative)

Figure 3.3.1 Average human face in grey-scale

The grey-scale differences, which are invariant across all the sample faces are strikingly apparent. The eye-eyebrow area seem to always contain dark intensity (low) gray-levels while nose forehead and cheeks contain bright intensity (high) grey-levels. After a great deal of experimentation, the researcher found that the following areas of the human face were suitable for a face detection system based on image invariants and a deformable template.



scaled colormap scaled colormap (negative)

Fig 5.3.2 Area chosen for face detection (indicated on average human face in gray scale)

The above facial area performs well as a basis for a face template, probably because of the clear divisions of the bright intensity invariant area by the dark intensity invariant regions. Once this pixel area is located by the face detection system, any particular area required can be segmented based on the proportions of the average human face After studying the above images it was subjectively decided by the author to use the following as a basis for dark intensity sensitive and bright intensity sensitive templates. Once these are located in a subject's face, a pixel area 33.3% (of the width of the square window) below this.

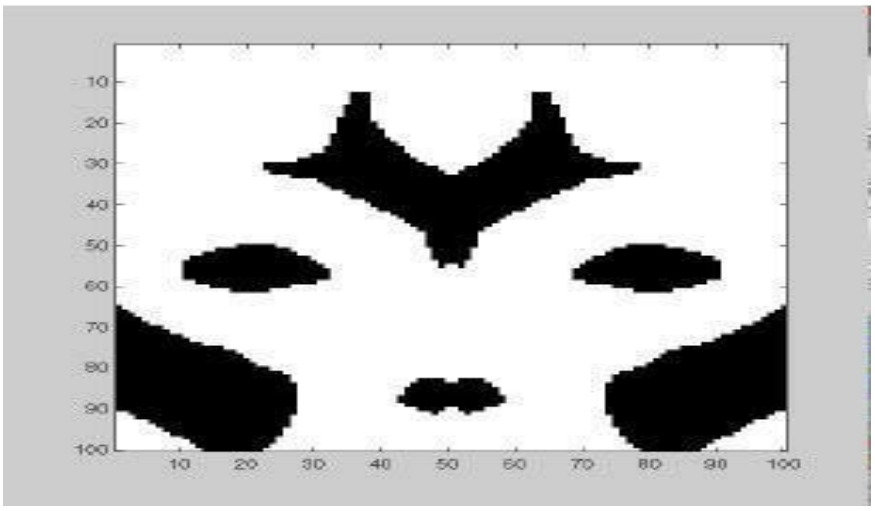
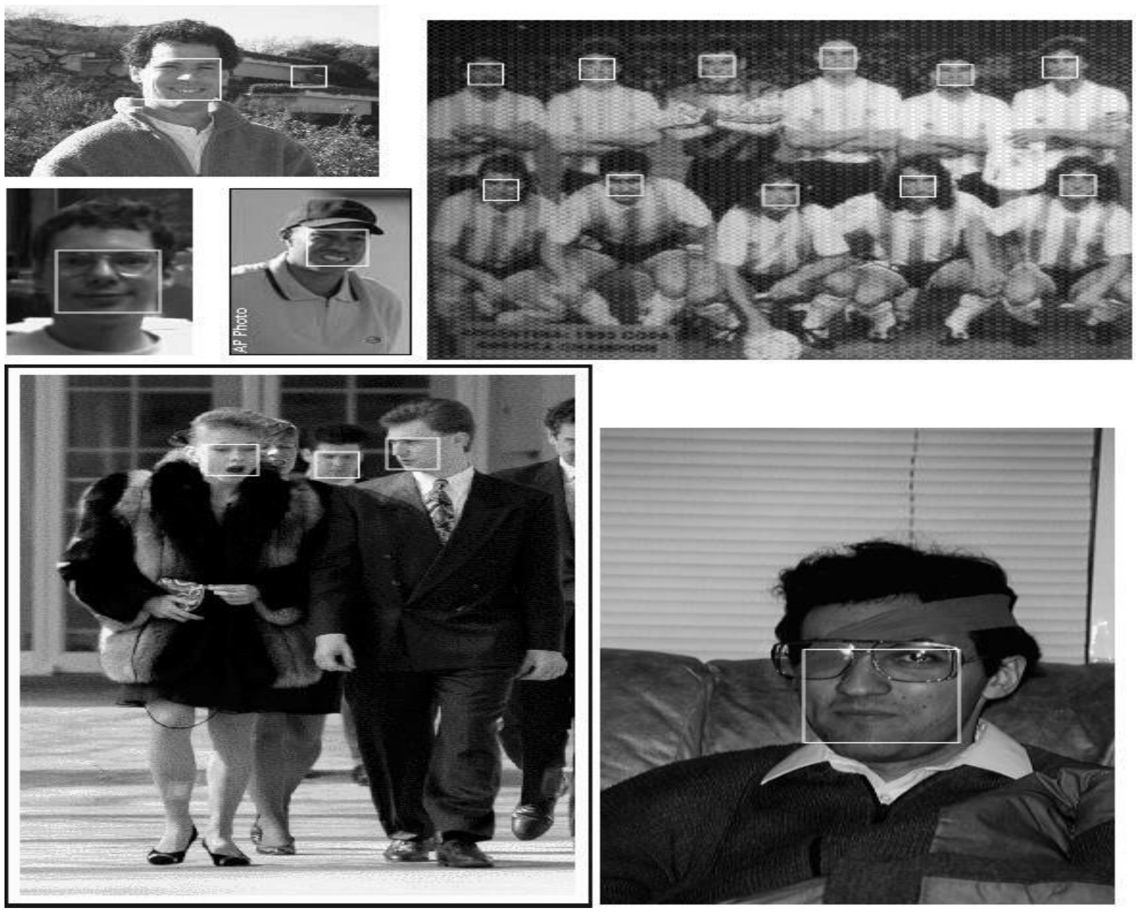


Figure 3.3.3: Basis for a bright intensity invariant sensitive template.

Note the slight differences which were made to the bright intensity invariant sensitive template (compare Figures 3.4 and 3.2) which were needed because of the pre-processing done by the system to overcome irregular lighting (chapter six). Now that a suitable dark and bright intensity invariant templates have been decided on, it is necessary to find a way of using these to make 2 A- units for a perceptron, i.e. a computational model is needed to assign neurons to the distributions displayed .



Fig

3.3.4 Scaned image detection

- San window over image
- Clasify window as either
    1. Face
    2. Non – Face

3.4 FACE DETECTION ALGORITHM

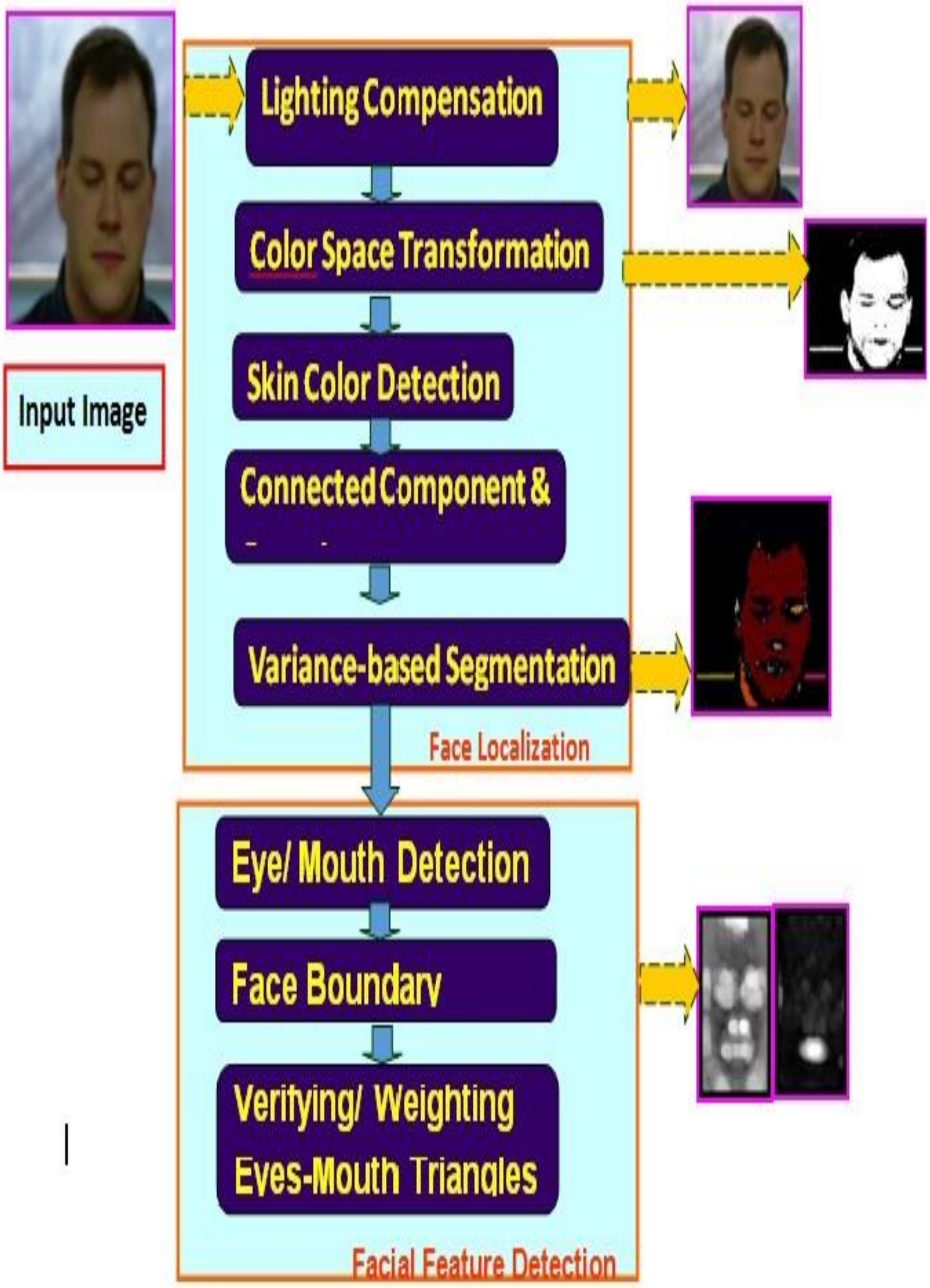
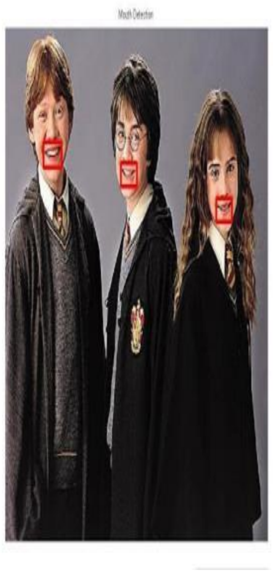


Fig 3.4 Face detection algorithm



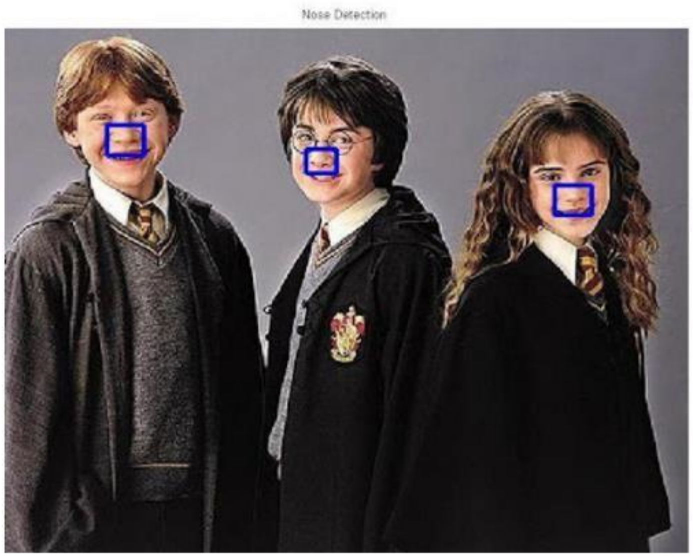
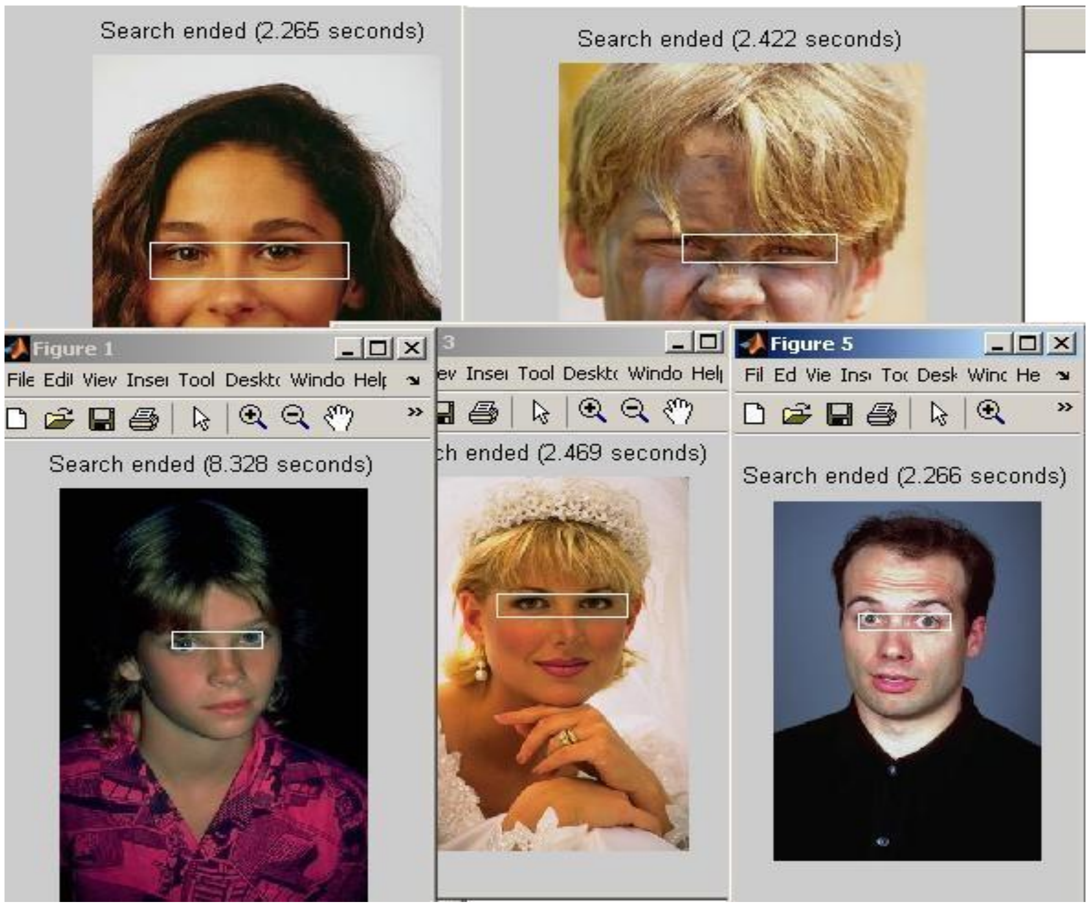


Fig 3.4.1 mouth detection

Fig 3.4.2 Noise detection



```
Search ended (2.265 seconds)
Search ended (2.422 seconds)
Search ended (8.328 seconds)
ch ended (2.469 seconds)
Search ended (2.266 seconds)
```

Fig 3.4.3 Eye detection

CHAPTER 4
FACE RECOGNITION

Over the last few decades many techniques have been proposed for face recognition. Many of the techniques proposed during the early stages of computer vision cannot be considered successful, but almost all of the recent approaches to the face recognition problem have been creditable. According to the research by Brunelli and Poggio (1993) all approaches to human face recognition can be divided into two strategies:

(1) Geometrical features and

(2) Template matching.

4.1 FACE RECOGNITION USING GEOMETRICAL FEATURES

This technique involves computation of a set of geometrical features such as nose width and length, mouth position and chin shape, etc. from the picture of the face we want to recognize. This set of features is then matched with the features of known individuals. A suitable metric such as Euclidean distance (finding the closest vector) can be used to find the closest match. Most pioneering work in face recognition was done using geometric features (Kanade, 1973), although Craw et al. (1987) did relatively recent work in this area.

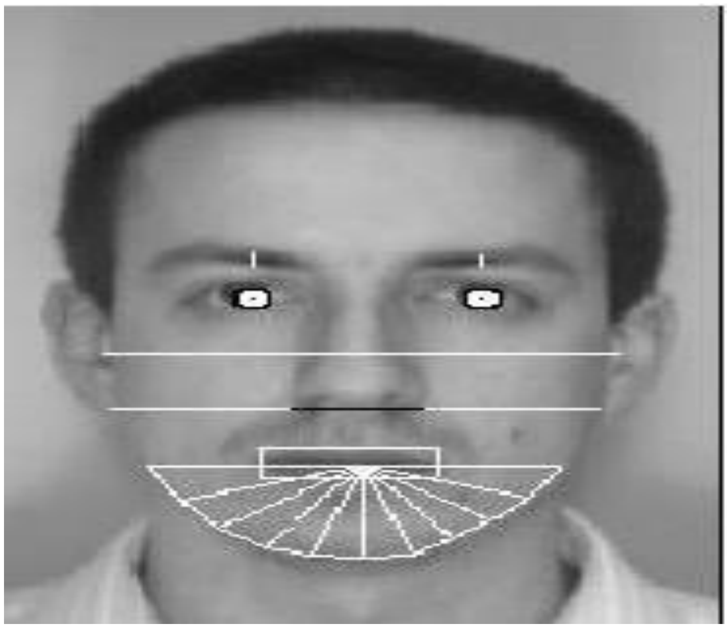


Fig 4.1 Geometrical features (white) which could be used for face recognition

The advantage of using geometrical features as a basis for face recognition is that recognition is possible even at very low resolutions and with noisy images (images with many disorderly pixel intensities). Although the face cannot be viewed in detail its overall geometrical configuration can be extracted for face recognition. The technique's main disadvantage is that automated extraction of the facial geometrical features is very hard. Automated geometrical feature extraction based recognition is also very sensitive to the scaling and rotation of a face in the image plane (Brunelli and Poggio, 1993). This is apparent when we examine Kanade's(1973) results where he reported a recognition rate of between 45-75 % with a database of only 20 people. However if these features are extracted manually as in Goldstein et al. (1971), and Kaya and Kobayashi (1972) satisfactory results may be obtained.

4.1.1 Face Recognition using Template Matching

This is similar the template matching technique used in face detection, except here we are not trying to classify an image as a 'face' or 'non-face' but are trying to recognize a face.

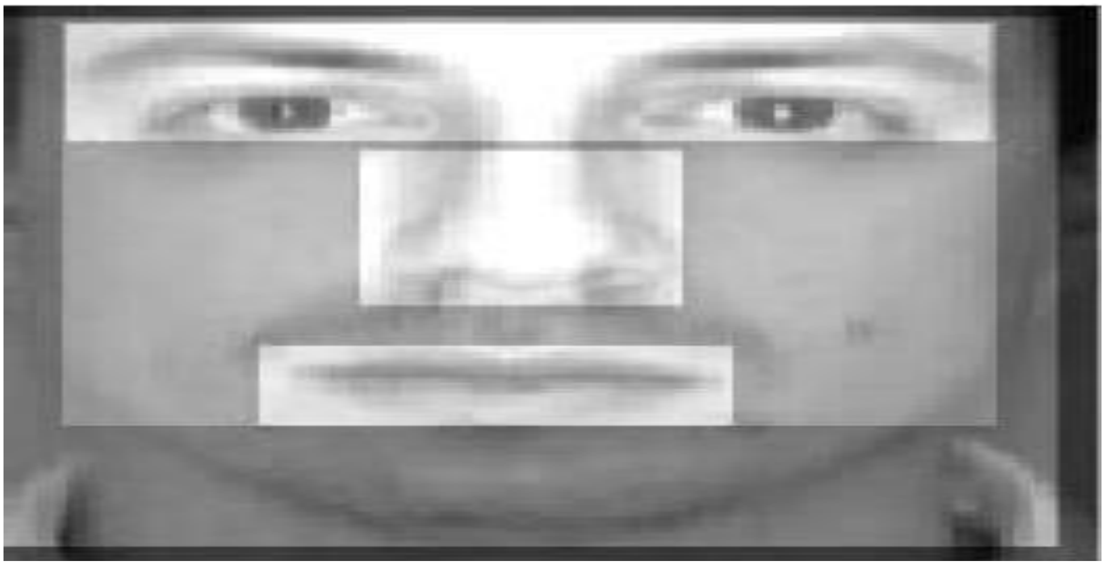


Fig 4.1.1: Face recognition using template matching

Whole face, eyes, nose and mouth regions which could be used in a template matching strategy. The basis of the template matching strategy is to extract whole facial regions (matrix of pixels) and compare these with the stored images of known individuals. Once again Euclidean distance can be used to find the closest match. The simple technique of comparing grey-scale intensity values for face recognition was used by Baron (1981). However there are far more sophisticated methods of template matching for face recognition. These involve extensive pre-processing and transformation of the extracted grey-level intensity values. For example, Turk and Pentland (1991a) used Principal Component Analysis, sometimes known as the eigenfaces approach, to pre-process the grey-levels and Wiskott et al. (1997) used Elastic Graphs encoded using Gabor filters to pre-process the extracted regions. An investigation of geometrical features versus template matching for face recognition by Brunelli and Poggio (1993) came to the conclusion that although a feature based strategy may offer higher recognition speed and smaller memory requirements, template based techniques offer superior recognition accuracy.

4.2 PROBLEM SCOP AND SYSTEM SPECIFICATION

The following problem scope for this project was arrived at after reviewing the literature on face detection and face recognition, and determining possible real-world situations where such systems would be of use. The following system(s) requirements were identified

1 A system to detect frontal view faces in static images

2 A system to recognize a given frontal view face

3 Only expressionless, frontal view faces will be presented to the face detection recognition 4 All implemented systems must display a high degree of lighting invariancy.

5 All systems must posses near real-time performance.

6 Both fully automated and manual face detection must be supported

7 Frontal view face recognition will be realised using only a single known image

8 Automated face detection and recognition systems should be combined into a fully automated face detection and recognition system. The face recognition sub-system must display a slight degree of invariency to scaling and rotation errors in the segmented image extracted by the face detection sub-system.

9 The frontal view face recognition system should be extended to a pose invariant face recognition system.

Unfortunately although we may specify constricting conditions to our problem domain, it may not be possible to strictly adhere to these conditions when implementing a system in the real-world.

4.3 BRIEF OUT LINE OF THE IMPLEMENTED SYSTEM

Fully automated face detection of frontal view faces is implemented using a deformable template algorithm relying on the image invariants of human faces. This was chosen because a similar neural-network based face detection model would have needed far too much training data to be implemented and would have used a great deal of computing time. The main difficulties in implementing a deformable template based technique were the creation of the bright and dark intensity sensitive templates and designing an efficient implementation of the detection algorithm.

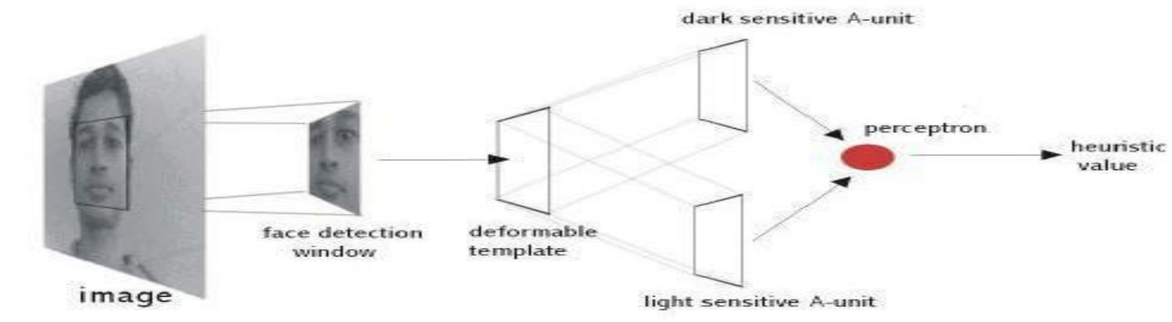


Fig 4.3 Implemented fully automated frontal view face detection model

A manual face detection system was realised by measuring the facial proportions of the average face, calculated from 30 test subjects. To detect a face, a human operator would identify the locations of the subject's eyes in an image and using the proportions of the average face, the system would segment an area from the image

A template matching based technique was implemented for face recognition. This was because of its increased recognition accuracy when compared to geometrical features based techniques and the fact that an automated geometrical features based technique would have required complex feature detection pre-processing.

Of the many possible template matching techniques, Principal Component Analysis was chosen because it has proved to be a highly robust in pattern recognition tasks and because it is relatively simple to implement. The author would also liked to have implemented a technique based on Elastic Graphs but could not find sufficient literature about the model to implement such a system during the limited time available for this project.

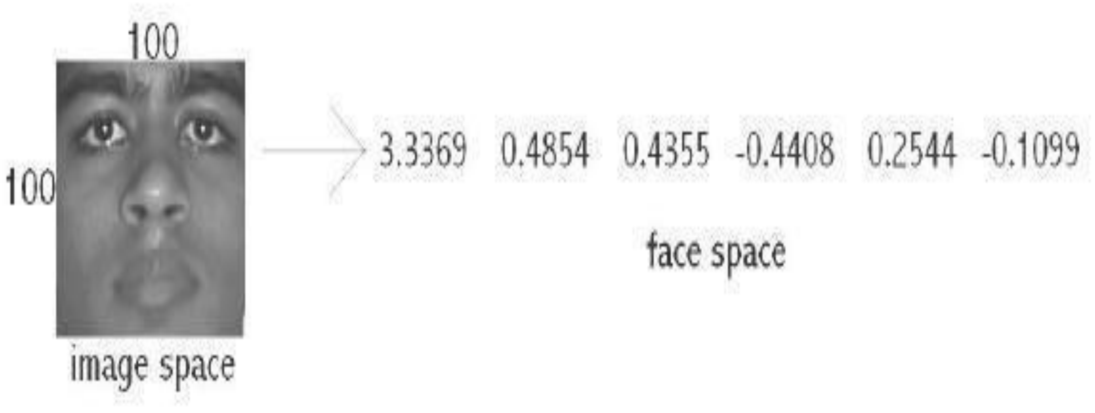


Fig 4.3.1: Principal Component Analysis transform from 'image space' to 'face space'.

Using Principal Component Analysis, the segmented frontal view face image is transformed from what is sometimes called 'image space' to 'face space'. All faces in the face database are transformed into face space. Then face recognition is achieved by transforming any given test image into face space and comparing it with the training set vectors. The closest matching training set vector should belong to the same individual as the test image. Principal Component Analysis is of special interest because the transformation to face space is based on the variation of human faces (in the training set). The values of the 'face space' vector correspond to the amount certain 'variations' are present in the test image

Face recognition and detection system is a pattern recognition approach for personal identification purposes in addition to other biometric approaches such as fingerprint recognition, signature, retina and so forth. Face is the most common biometric used by humans applications ranges from static, mug-shot verification in a cluttered background.

Fig 4.3.2 Face Recognition

4.4 FACE RECOGNITION DIFFICULTIES

1. Identify similar faces (inter-class similarity)

2. Accommodate intra-class variability due to

2.1 head pose

2.2 illumination conditions

2.3 expressions

2.4 facial accessories

2.5 aging effects

3. Cartoon faces

4.4.1 Inter - class similarity:

- Different persons may have very similar appearance

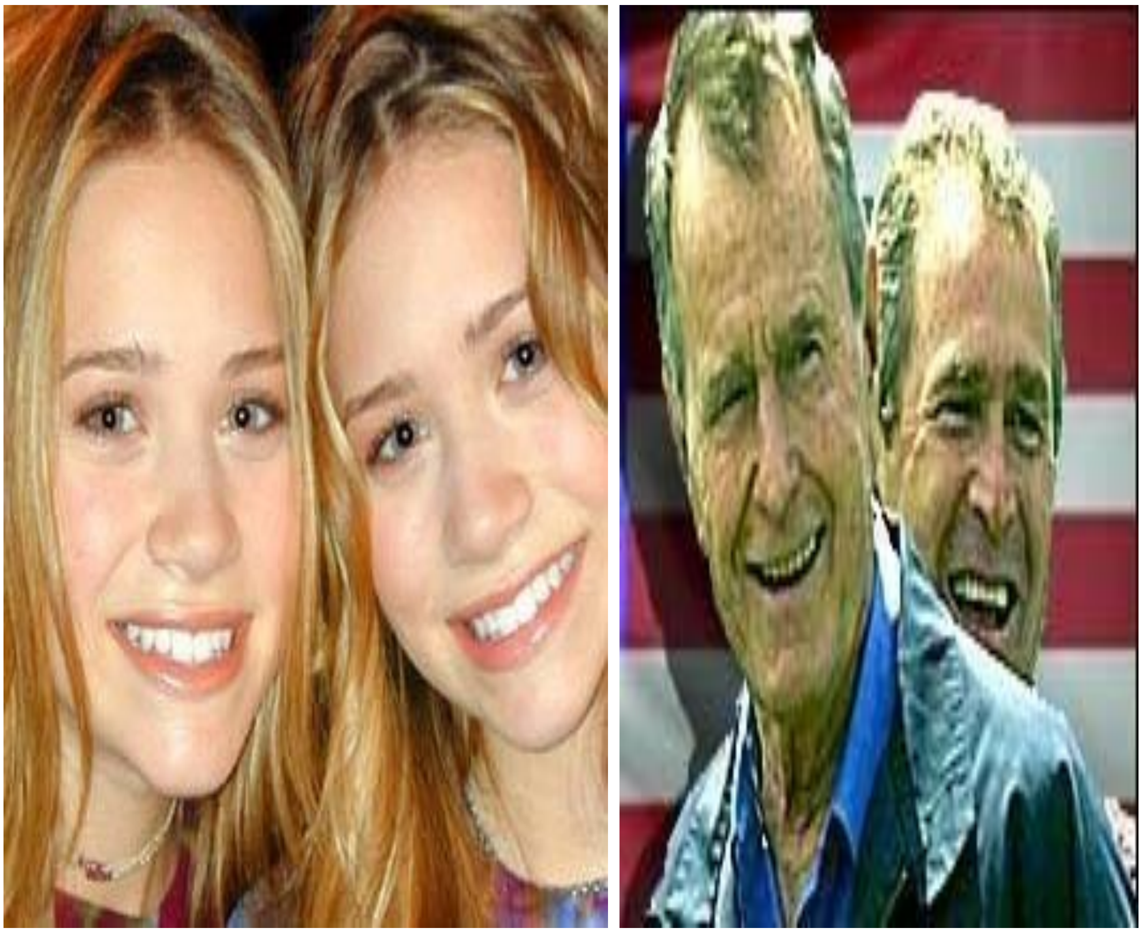


Fig 6.4.1 Face recognition twins and other and Son

Face recognition and detection system is a pattern recognition approach for personal identification purposes in addition to other biometric approaches such as fingerprint recognition, signature, retina and so forth. The variability in the faces, the images are processed before they are fed into the network. All positive examples that is the face images are obtained by cropping images with frontal faces to include only the front view. All the cropped images are then corrected for lighting through standard algorithms.

4.4.2 Inter Class Variability

Faces with intra-subject variations in pose, illumination, expression, accessories, color, occlusions, and brightnes .

4.5 PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal Component Analysis (or Karhunen-Loeve expansion) is a suitable strategy for face recognition because it identifies variability between human faces, which may not be immediately obvious. Principal Component Analysis (hereafter PCA) does not attempt to categorise faces using familiar geometrical differences, such as nose length or eyebrow width. Instead, a set of human faces is analysed using PCA to determine which 'variables' account for the variance of faces. In face recognition, these variables are called eigen faces because when plotted they display an eerie resemblance to human faces. Although PCA is used extensively in statistical analysis, the pattern recognition community started to use PCA for classification only relatively recently. As described by Johnson and Wichern (1992), 'principal component analysis is concerned with explaining the variance- covariance structure through a few linear combinations of the original variables.' Perhaps PCA's greatest strengths are in its ability for data reduction and interpretation. For example a 100x100 pixel area containing a face can be very accurately represented by just 40 eigen values. Each eigen value describes the magnitude of each eigen face in each image. Furthermore, all interpretation (i.e. recognition) operations can now be done using just the 40 eigen values to represent a face instead of the manipulating the 10000 values contained in a 100x100 image. Not only is this computationally less demanding but the fact that the recognition information of several thousand.

4.6 UNDERSTANDING EIGENFACES

Any grey scale face image I(x,y) consisting of a NxN array of intensity values may also be consider as a vector of N^2. For example, a typical 100x100 image used in this thesis will have to be transformed into a 10000 dimension vector!

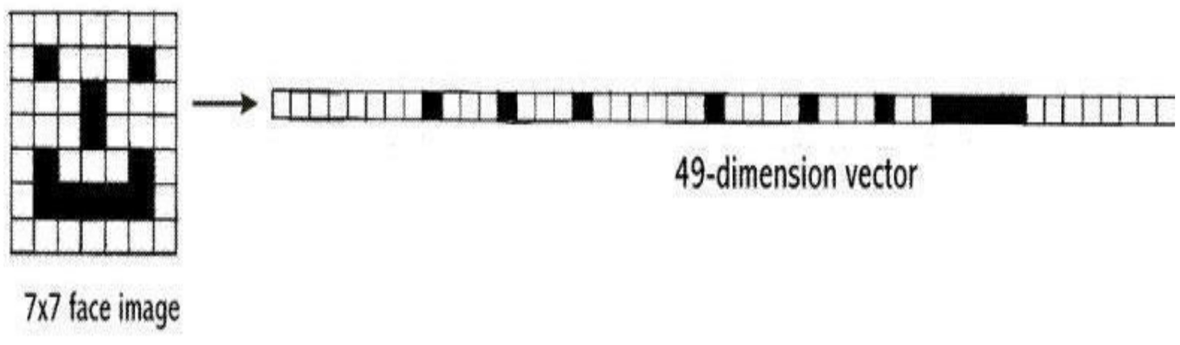
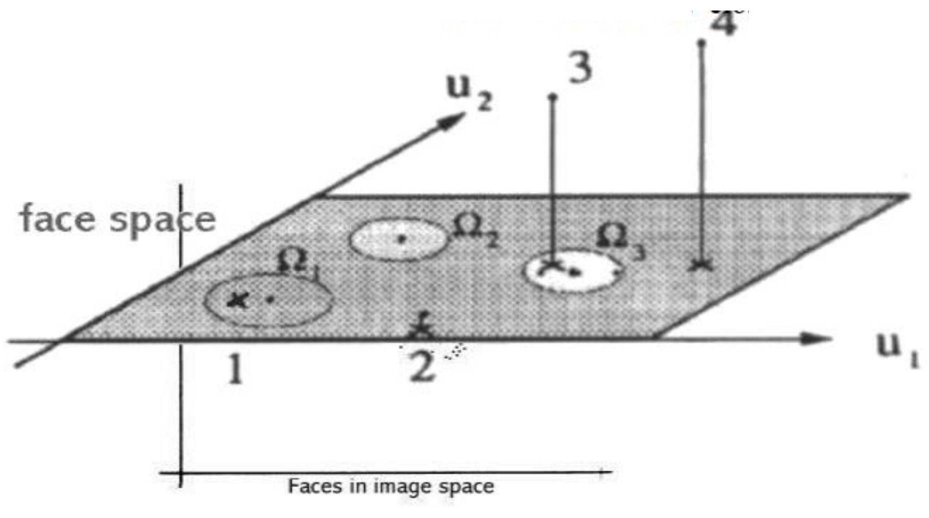


Fig 4.6.0 A 7x7 face image transformed into a 49 dimension vector

This vector can also be regarded as a point in 10000 dimension space. Therefore, all the images of subjects' whose faces are to be recognized can be regarded as points in 10000 dimension space. Face recognition using these images is doomed to failure because all human face images are quite similar to one another so all associated vectors are very close to each other in the 10000- dimension space.



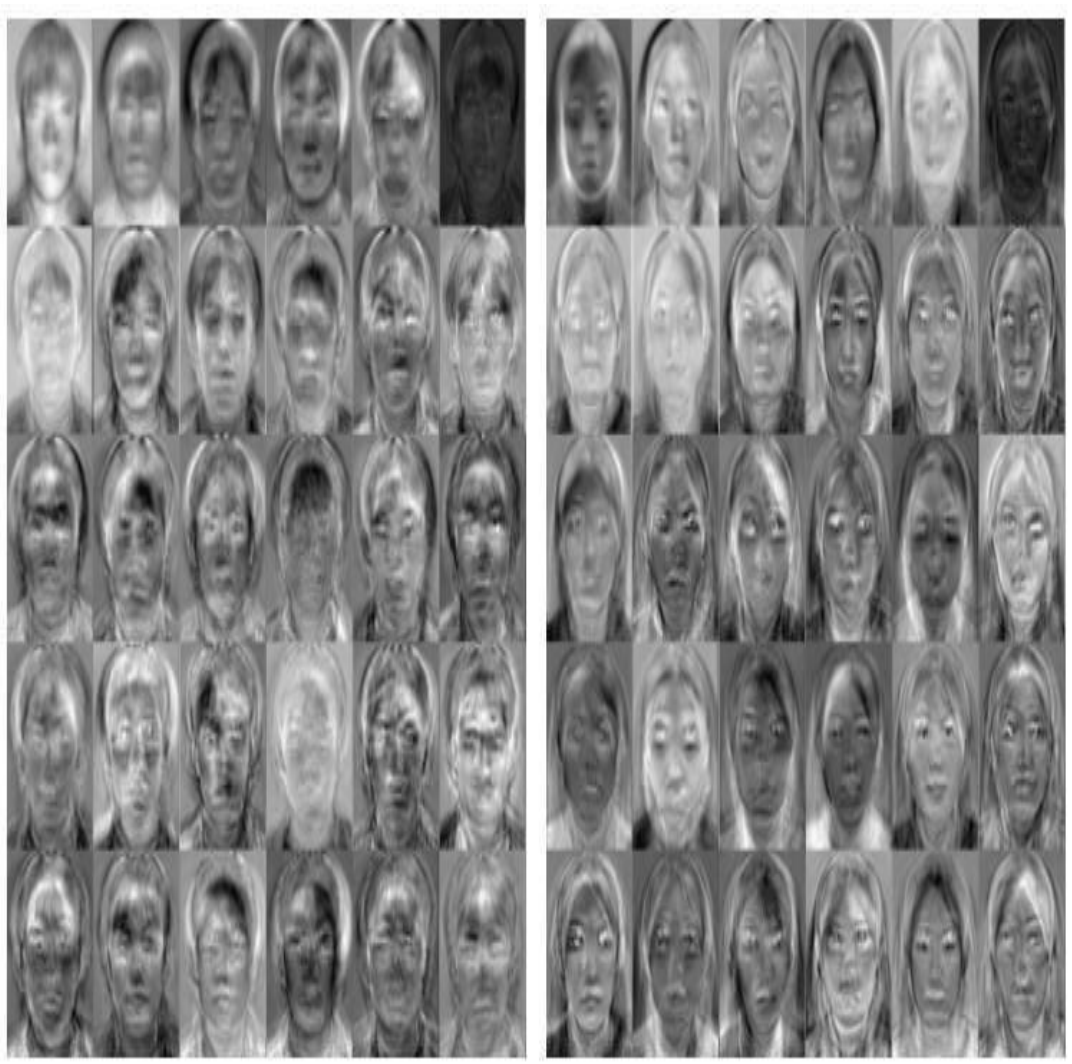


Fig 4.6.1 Eigenfaces

The transformation of a face from image space (I) to face space (f) involves just a simple matrix multiplication. If the average face image is A and U contains the (previously calculated) eigenfaces,

$$f = U * (I - A)$$

This is done to all the face images in the face database (database with known faces) and to the image (face of the subject) which must be recognized. The possible results when projecting a face into face space are given in the following figure.

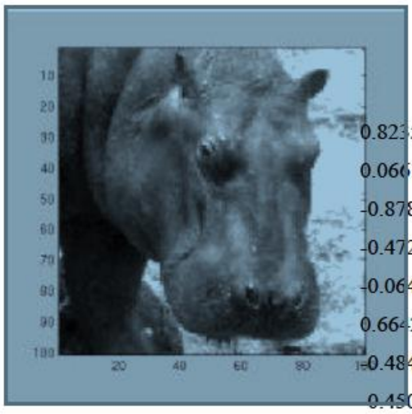
There are four possibilities:

1. Projected image is a face and is transformed near a face in the face database
2.Projected image is a face and is not transformed near a face in the face database
3.Projected image is not a face and is transformed near a face in the face database
4. Projected image is not a face and is not transformed near a face in the face database

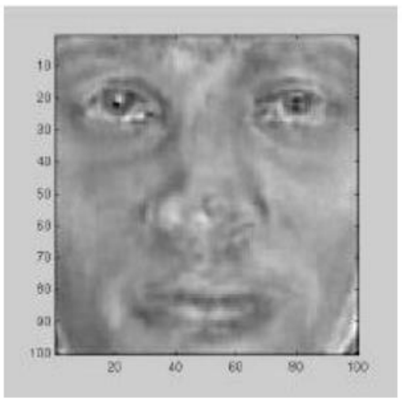
While it is possible to find the closest known face to the transformed image face by calculating the Euclidean distance to the other vectors, how does one know whether the image that is being transformed actually contains a face? Since PCA is a many-to-one transform, several vectors in the image space (images) will map to a point in face space (the problem is that even non-face images may transform near a known face image's faces space vector). Turk and Pentland (1991a), described a simple way of checking whether an image is actually of a face. This is by transforming an image into face space and then transforming it back (reconstructing) into image space. Using the previous notation,

$$I' = UT * U * (I - A)$$

With these calculations it is possible to verify that an image is of a face and recognise that face. O'Toole et al. (1993) did some interesting work on the importance of eigen faces with large and small eigenvalues. They showed that the eigen vectors with larger eigenvalues convey information relative to the basic shape and structure of the faces. This kind of information is most useful in categorising faces according to sex, race etc. Eigen vectors with smaller eigenvalues tend to capture information that is specific to single or small subsets of learned faces and are useful for distinguishing a particular face from any other face. Turk and Pentland (1991a) showed that about 40 eigen faces were sufficient for a very good description of human faces since the reconstructed image have only about 2% RMS. pixel-by-pixel errors.

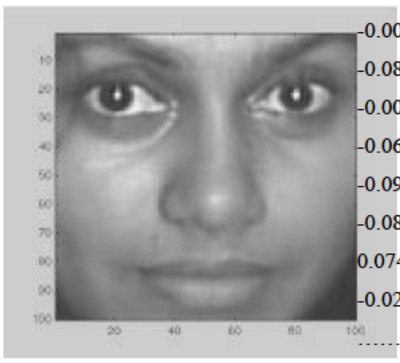


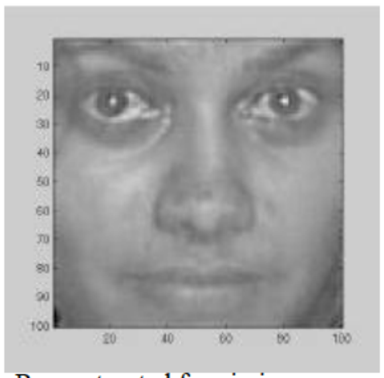
0.8235
0.0661
-0.8786
-0.4727
-0.0646
0.6642
-0.4840
-0.4501
-0.2506
0.1591
0.3359
0.0048
0.0745
....



Hippo in image space Hippo in face space Reconstructed hippo in image space

0.7253
-0.0392
0.2896
-0.1725
-0.2642
-0.0014
-0.0814
-0.0054
-0.0623
-0.0965
-0.0879
0.0745
-0.0261
....





Face in image space Face in face space Reconstructed face in image space

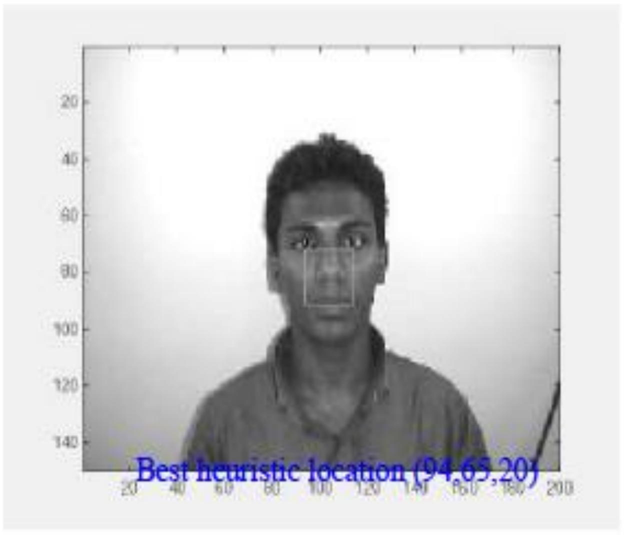
Fig 4.6.3 Images and there reconstruction. The Euclidean distance between a face image and its reconstruction will be lower than that of a non-face image

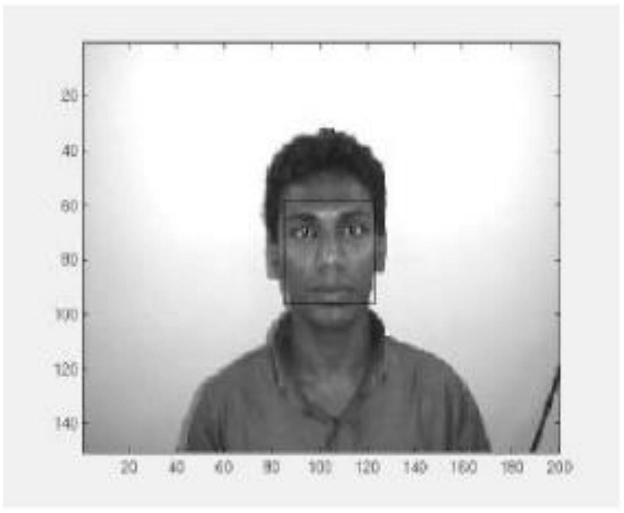
4.7 IMPROVING FACE DETECTION USING RECONSTRUCTION

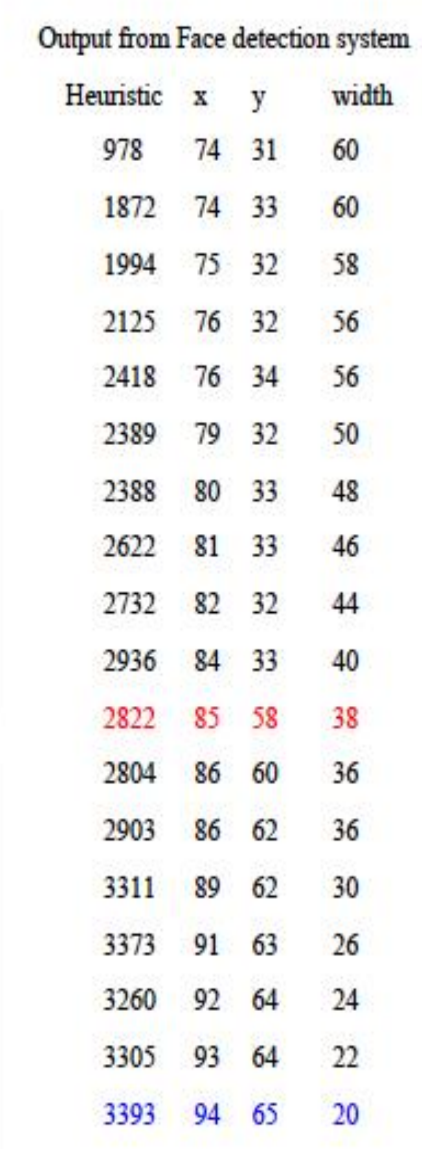
Reconstruction cannot be used as a means of face detection in images in near real-time since it would involve resizing the face detection *window* area and large matrix multiplication, both of which are computationally expensive. However, reconstruction can be used to verify whether potential face locations identified by the deformable template algorithm actually contain a face. If the reconstructed image differs greatly from the face detection *window* then the *window* probably does not contain a face. Instead of just identifying a single potential face location, the face detection algorithm can be modified to output many high 'faceness' locations which can be verified using reconstruction. This is especially useful because occasionally the best 'faceness' location found by the deformable template algorithm may not contain the ideal frontal view face pixel area.

potential face locations that have been identified by the face detection system (the best face locations it found on its search) are checked whether they contain a face. If the threshold level (maximum difference between reconstruction and original for the original to be a face) is set correctly this will be an efficient way to detect a face. The deformable template algorithm is fast and can reduce the search space of potential face locations to a handful of positions. These are then checked using reconstruction. The number of locations found by the face detection system can be changed by getting it to output, not just the best face locations it has found so far but any location, which has a 'faceness' value, which for example is, at least 0.9 times the best heuristic value that has been found so far.

Then there will be many more potential face locations to be checked using reconstruction. This and similar speed versus accuracy trade-off decisions have to be made keeping in mind the platform on which the system is implemented.







Output from Face detection system

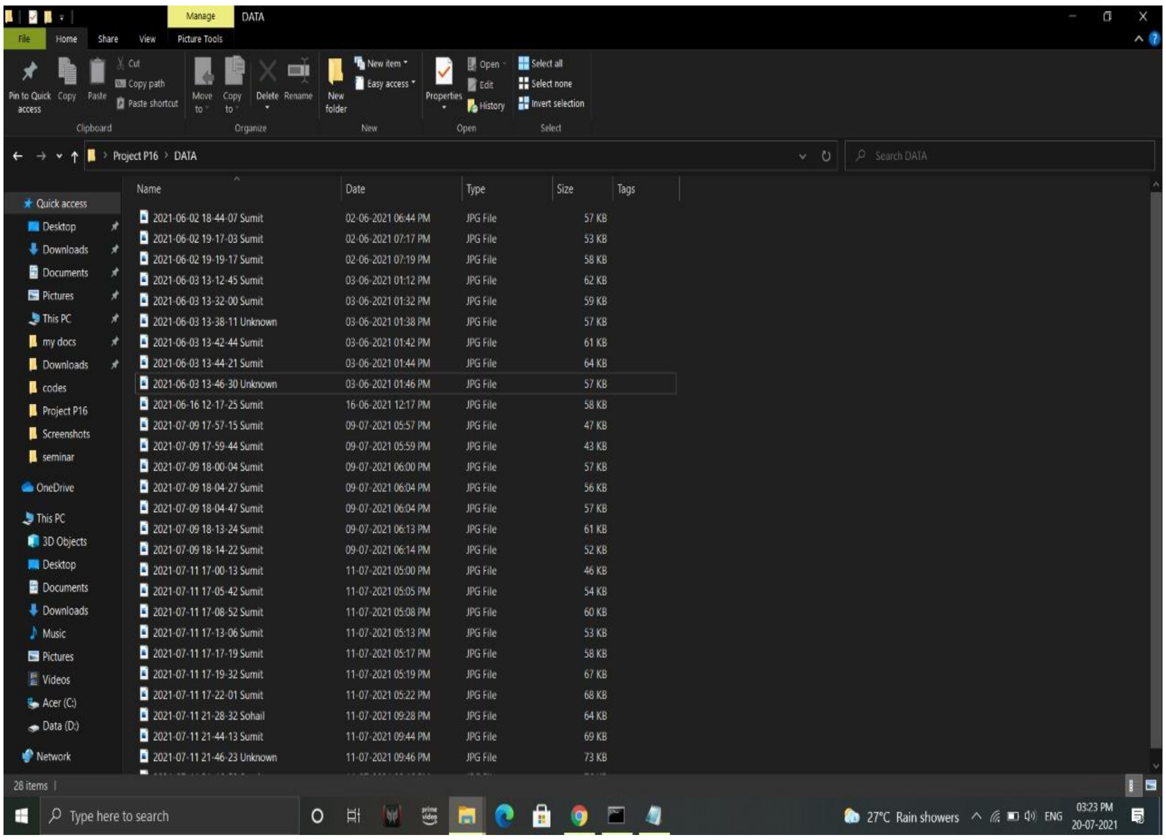
Heuristic	x	y	width
978	74	31	60
1872	74	33	60
1994	75	32	58
2125	76	32	56
2418	76	34	56
2389	79	32	50
2388	80	33	48
2622	81	33	46
2732	82	32	44
2936	84	33	40
2822	85	58	38
2804	86	60	36
2903	86	62	36
3311	89	62	30
3373	91	63	26
3260	92	64	24
3305	93	64	22
3393	94	65	20

Similarly, instead of using reconstruction to check the face detection system's output, the output's correlation with the average face can be checked. The segmented areas with a high correlation probably contains a face. Once again a threshold value will have to be established to classify faces from non-faces. Similar to reconstruction, resizing the segmented area and calculating its correlation with the average face is far too expensive to be used alone for face detection but is suitable for verifying the output of the face detection system.

4.8 POSE INVARIANT FACE RECOGNITION

Extending the frontal view face recognition system to a pose-invariant recognition system is quite simple if one of the proposed specifications of the face recognition system is relaxed. Successful pose-invariant recognition will be possible if many images of a known individual are in the face database. Nine images from each known individual can be taken as shown below. Then if an image of the same individual is submitted within a 30^{o} angle from the frontal view he or she can be identified.

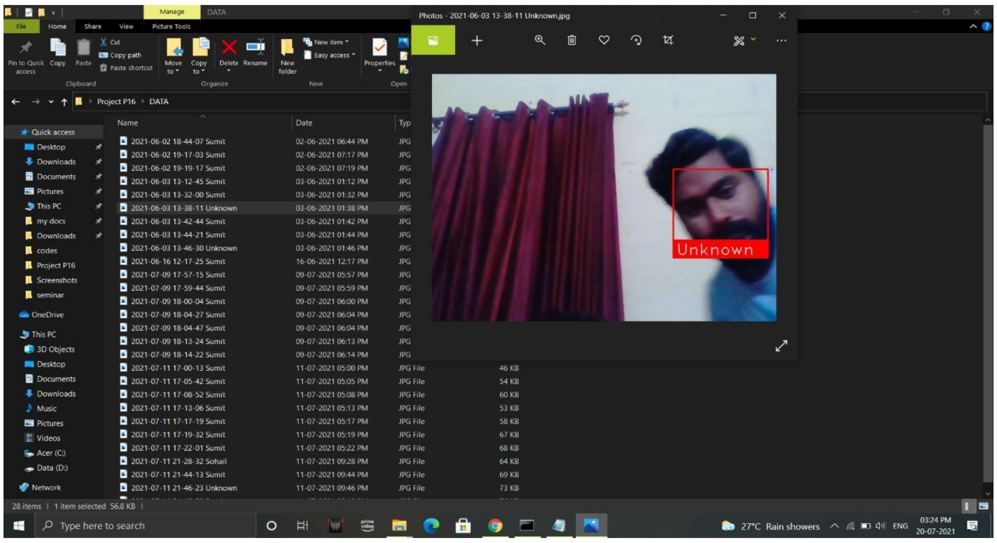
Face database from a single known individual



Name	Date	Type	Size	Tags
2021-06-02 18-44-07 Sumit	02-06-2021 06:44 PM	JPG File	57 KB	
2021-06-02 19-17-03 Sumit	02-06-2021 07:17 PM	JPG File	53 KB	
2021-06-02 19-19-17 Sumit	02-06-2021 07:19 PM	JPG File	58 KB	
2021-06-03 13-12-45 Sumit	03-06-2021 01:12 PM	JPG File	62 KB	
2021-06-03 13-32-00 Sumit	03-06-2021 01:32 PM	JPG File	59 KB	
2021-06-03 13-38-11 Unknown	03-06-2021 01:38 PM	JPG File	57 KB	
2021-06-03 13-42-44 Sumit	03-06-2021 01:42 PM	JPG File	61 KB	
2021-06-03 13-44-21 Sumit	03-06-2021 01:44 PM	JPG File	64 KB	
2021-06-03 13-46-30 Unknown	03-06-2021 01:46 PM	JPG File	57 KB	
2021-06-16 12-17-25 Sumit	16-06-2021 12:17 PM	JPG File	58 KB	
2021-07-09 17-57-15 Sumit	09-07-2021 05:57 PM	JPG File	47 KB	
2021-07-09 17-59-44 Sumit	09-07-2021 05:59 PM	JPG File	43 KB	
2021-07-09 18-00-04 Sumit	09-07-2021 06:00 PM	JPG File	57 KB	
2021-07-09 18-04-27 Sumit	09-07-2021 06:04 PM	JPG File	56 KB	
2021-07-09 18-04-47 Sumit	09-07-2021 06:04 PM	JPG File	57 KB	
2021-07-09 18-13-24 Sumit	09-07-2021 06:13 PM	JPG File	61 KB	
2021-07-09 18-14-22 Sumit	09-07-2021 06:14 PM	JPG File	52 KB	
2021-07-11 17-00-13 Sumit	11-07-2021 05:00 PM	JPG File	46 KB	
2021-07-11 17-05-42 Sumit	11-07-2021 05:05 PM	JPG File	54 KB	
2021-07-11 17-08-52 Sumit	11-07-2021 05:08 PM	JPG File	60 KB	
2021-07-11 17-13-06 Sumit	11-07-2021 05:13 PM	JPG File	53 KB	
2021-07-11 17-17-19 Sumit	11-07-2021 05:17 PM	JPG File	58 KB	
2021-07-11 17-19-32 Sumit	11-07-2021 05:19 PM	JPG File	67 KB	
2021-07-11 17-22-01 Sumit	11-07-2021 05:22 PM	JPG File	68 KB	
2021-07-11 21-28-32 Sohail	11-07-2021 09:28 PM	JPG File	64 KB	
2021-07-11 21-44-13 Sumit	11-07-2021 09:44 PM	JPG File	69 KB	
2021-07-11 21-46-23 Unknown	11-07-2021 09:46 PM	JPG File	73 KB	

Fig 4.8: Image Dataset

Unknown image from same individual to be identified



```
Photos - 2021-06-03 13-38-11 Unknown.jpg

Project P16 > DATA

Name                          Date                 Type
2021-06-02 18-44-07 Sumit     02-06-2021 06:44 PM  JPG
2021-06-02 19-17-03 Sumit     02-06-2021 07:17 PM  JPG
2021-06-02 19-19-17 Sumit     02-06-2021 07:19 PM  JPG
2021-06-03 13-12-45 Sumit     03-06-2021 01:12 PM  JPG
2021-06-03 13-32-00 Sumit     03-06-2021 01:32 PM  JPG
2021-06-03 13-38-11 Unknown   03-06-2021 01:38 PM  JPG
2021-06-03 13-42-44 Sumit     03-06-2021 01:42 PM  JPG
2021-06-03 13-44-21 Sumit     03-06-2021 01:44 PM  JPG
2021-06-03 13-46-30 Unknown   03-06-2021 01:46 PM  JPG
2021-06-16 12-17-25 Sumit     16-06-2021 12:17 PM  JPG
2021-07-09 17-57-15 Sumit     09-07-2021 05:57 PM  JPG
2021-07-09 17-59-44 Sumit     09-07-2021 05:59 PM  JPG
2021-07-09 18-00-04 Sumit     09-07-2021 06:00 PM  JPG
2021-07-09 18-04-27 Sumit     09-07-2021 06:04 PM  JPG
2021-07-09 18-04-47 Sumit     09-07-2021 06:04 PM  JPG
2021-07-09 18-13-24 Sumit     09-07-2021 06:13 PM  JPG
2021-07-09 18-14-22 Sumit     09-07-2021 06:14 PM  JPG
2021-07-11 17-00-13 Sumit     11-07-2021 05:00 PM  JPG File   46 KB
2021-07-11 17-05-42 Sumit     11-07-2021 05:05 PM  JPG File   54 KB
2021-07-11 17-08-52 Sumit     11-07-2021 05:08 PM  JPG File   60 KB
2021-07-11 17-13-06 Sumit     11-07-2021 05:13 PM  JPG File   53 KB
2021-07-11 17-17-19 Sumit     11-07-2021 05:17 PM  JPG File   58 KB
2021-07-11 17-19-32 Sumit     11-07-2021 05:19 PM  JPG File   67 KB
2021-07-11 17-22-01 Sumit     11-07-2021 05:22 PM  JPG File   68 KB
2021-07-11 21-28-32 Sohail    11-07-2021 09:28 PM  JPG File   64 KB
2021-07-11 21-44-13 Sumit     11-07-2021 09:44 PM  JPG File   69 KB
2021-07-11 21-46-23 Unknown   11-07-2021 09:46 PM  JPG File   73 KB

28 items | 1 item selected 56.8 KB |
```

Fig: 4.8.1: Pose invariant face recognition.

Pose invariant face recognition highlights the generalisation ability of PCA. For example, when an individual's frontal view and 30^o left view known, even the individual's 15^o left view can be recognised.

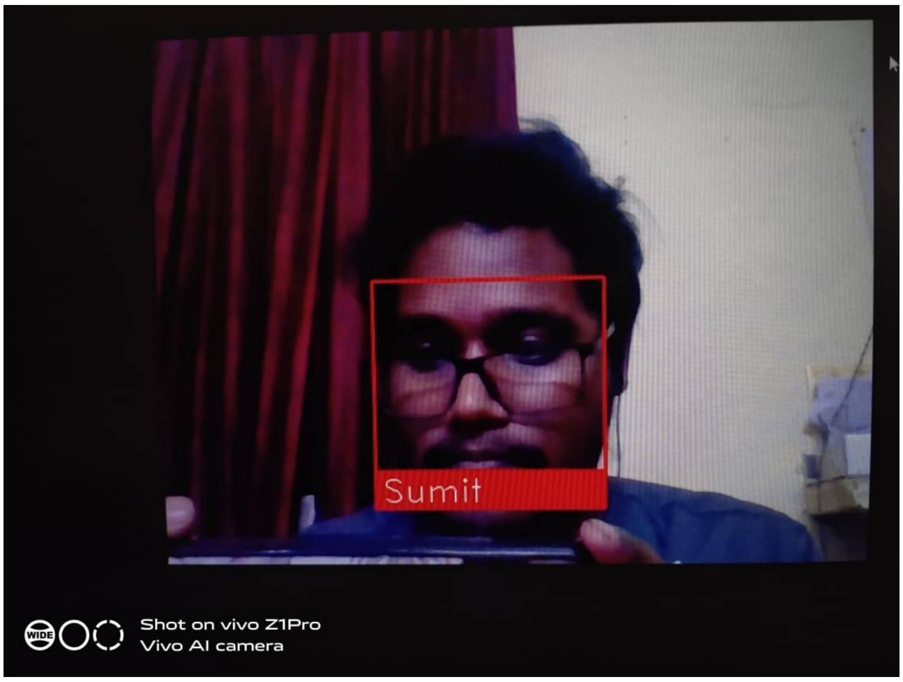


Fig 4.8.2 Face Recognised

CHAPTER 5
GUI AND OTP

5.1 GUI (GRAPHICAL USER INTERFACE)

Graphical User Interface (GUI) is nothing but a desktop application which helps you to interact with the computers. They perform different tasks in the desktops, laptops and other electronic devices.

- **GUI** apps like **Text-Editors** create, read, update and delete different types of files.
- Apps like Sudoku, Chess and Solitaire are games which you can play.
- **GUI** apps like **Google Chrome, Firefox and Microsoft Edge** browse through the **Internet**.

They are some different types of **GUI** apps which we daily use on the laptops or desktops. We are going to learn how to create those type of apps.

As this is an Introduction to GUI, make sure you stay tuned till the end as we will create a really simple and nice **GUI** app.

Well, it is a personal preference that I prefer GUI over command line. Not that there is something wrong with the command line but I prefer more intuitive and interactive applications with a lot of visuals.

5.1.1. TKINTER

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The name Tkinter comes from Tk interface.

Tkinter is the GUI library for Python. In Python, it is easy to create GUI applications with Tkinter. Tkinter is the object-oriented interface to the Tk, and can be used to create a graphical user interface application using the Tkinter function.

- An important feature in favor of Tkinter is that it is cross-platform, so the same code can easily work on Windows, macOS, and Linux.
- Tkinter is a lightweight module.
- It is simple to use.

You don’t need to worry about the installation of the Tkinter module separately as it comes with Python already. It gives an object-oriented interface to the Tk GUI toolkit. Graphical User Interface (GUI) is a form of user interface which allows users to interact with computers through visual indicators using items such as icons, menus, windows, etc. It has advantages over the Command Line Interface (CLI) where users interact with computers by writing commands using keyboard only and whose usage is more difficult than GUI.

Python has a lot of GUI frameworks, but Tkinter is the only framework that’s built into the Python standard library. Tkinter has several strengths. It’s cross-platform, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they’re run.

Although Tkinter is considered the de-facto Python GUI framework, it’s not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you’re looking for.

However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern sheen is unnecessary, and the top priority is to build something that’s functional and cross-platform quickly.

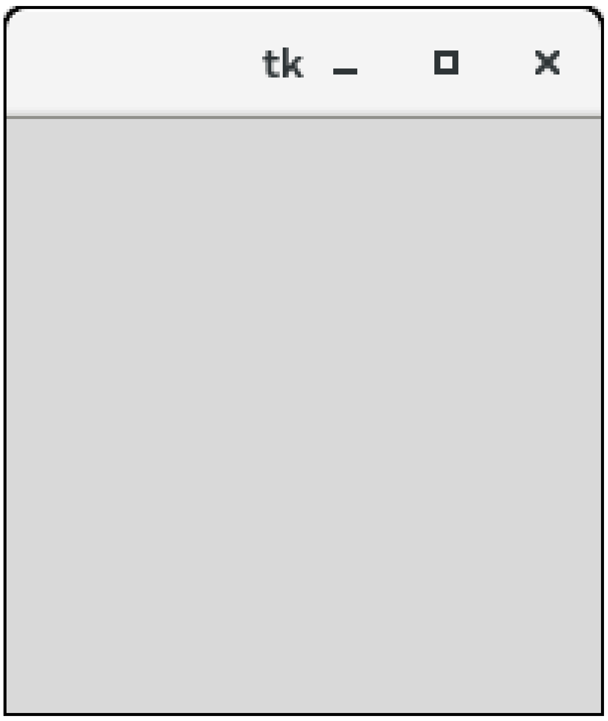


Fig 5.1.1: TKINTER INTERFACE

5.2 TKINTER DESCRIPTION

- Button-In Python, we use the button widget to display buttons in your application.
- Canvas-In Python, we use the canvas widget to draw shapes, such as lines, ovals, arcs in your application.
- Check button-In Python, we use the check button widget to display a number of options as checkboxes. The user can select multiple options at the same time.
- Entry-In Python, we use the entry widget to display a single-line text field for accepting values from a user.
- Frame-In Python, we use the frame widget as a container widget to organize other widgets.

- Label-In Python, we use the label widget to provide a single-line caption for other widgets. It can also contain images.
- Listbox-In Python, we use the listbox widget to provide a list of options to a user.
- Menubar-In Python, we use the menu button widget to display menus in your application. Menu-In Python, we use the menu widget to provide various commands to a user. These commands are contained inside Menubutton.
- Message-In Python, we use the message widget to display multiline text fields for accepting values from a user.
- Radio button-In Python, we use the radio Button widget to display a number of options as radio buttons.
- The user can select only one option at a one time.
- Scale-In Python,we use the scale widget to provide a slider widget.
- Scrollbar-In Python, we use the scrollbar widget to add scrolling capability to various widgets, such as list boxes.
- Text-In Python, we use the text widget to display text in multiple lines.
- Toplevel-In Python, we use the toplevel widget to provide a separate window container.
- Spinbox-In Python, we use the spinbox widget as a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values.
- PanedWindow-In Python, we use the panedwindow as a container widget that may contain any number of panes, arranged horizontally or vertically.
- Labelframe-In Python, we use labelframe as a simple container widget. The purpose is to act as a spacer or container for complex window layouts.
- Messagebox-Messagebox is used to display message boxes in your apps.

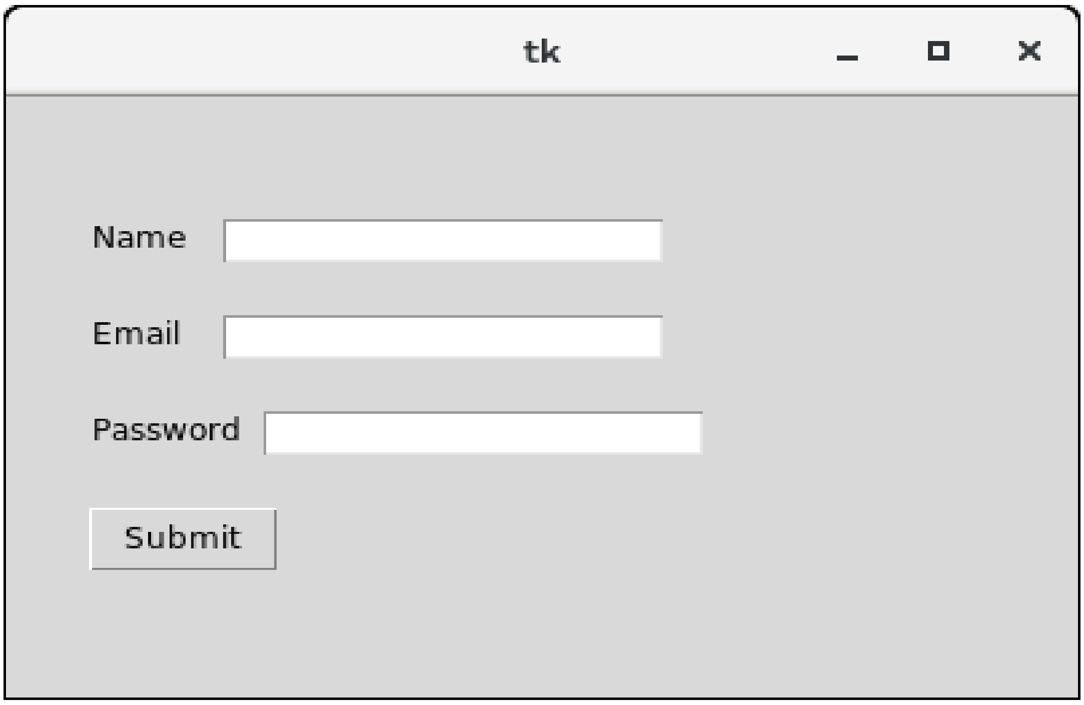


Fig 5.2 TKINTER INTERFACE WITH OPTIONS

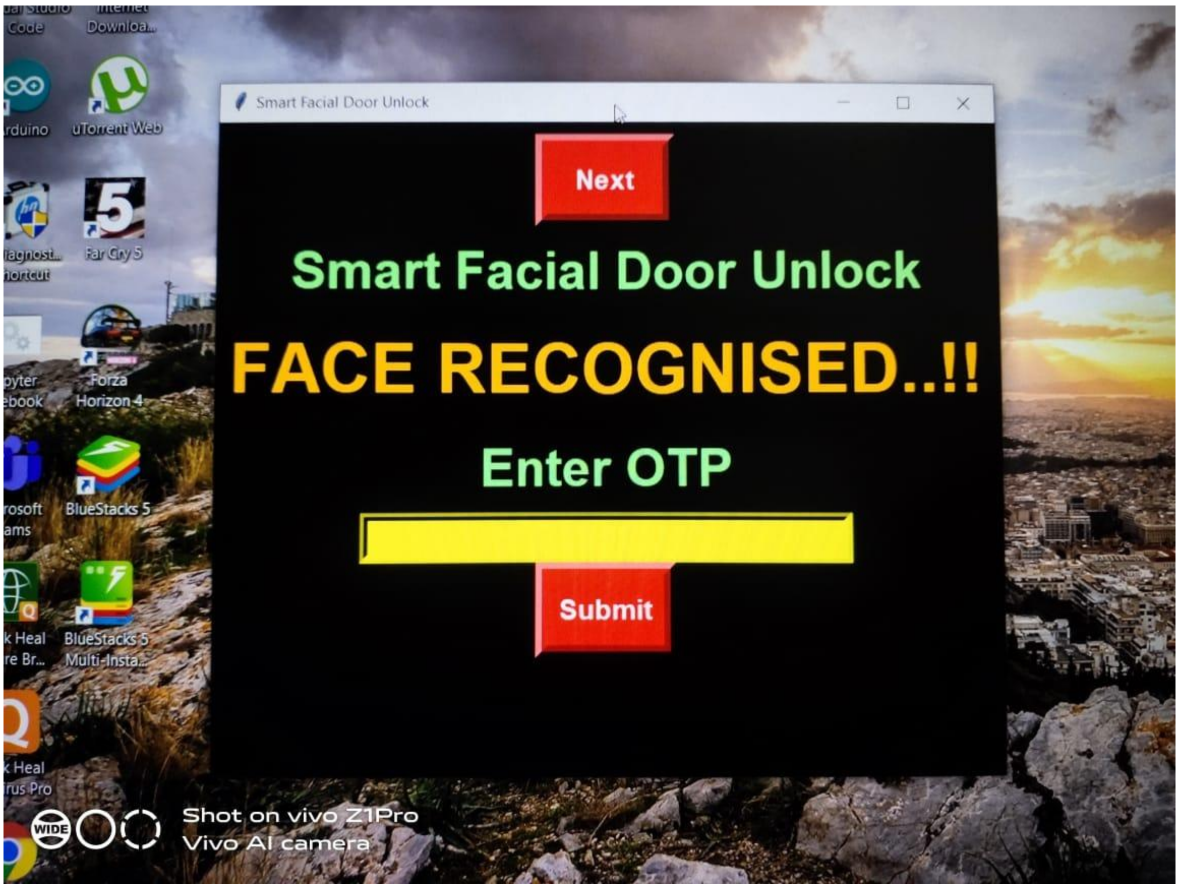


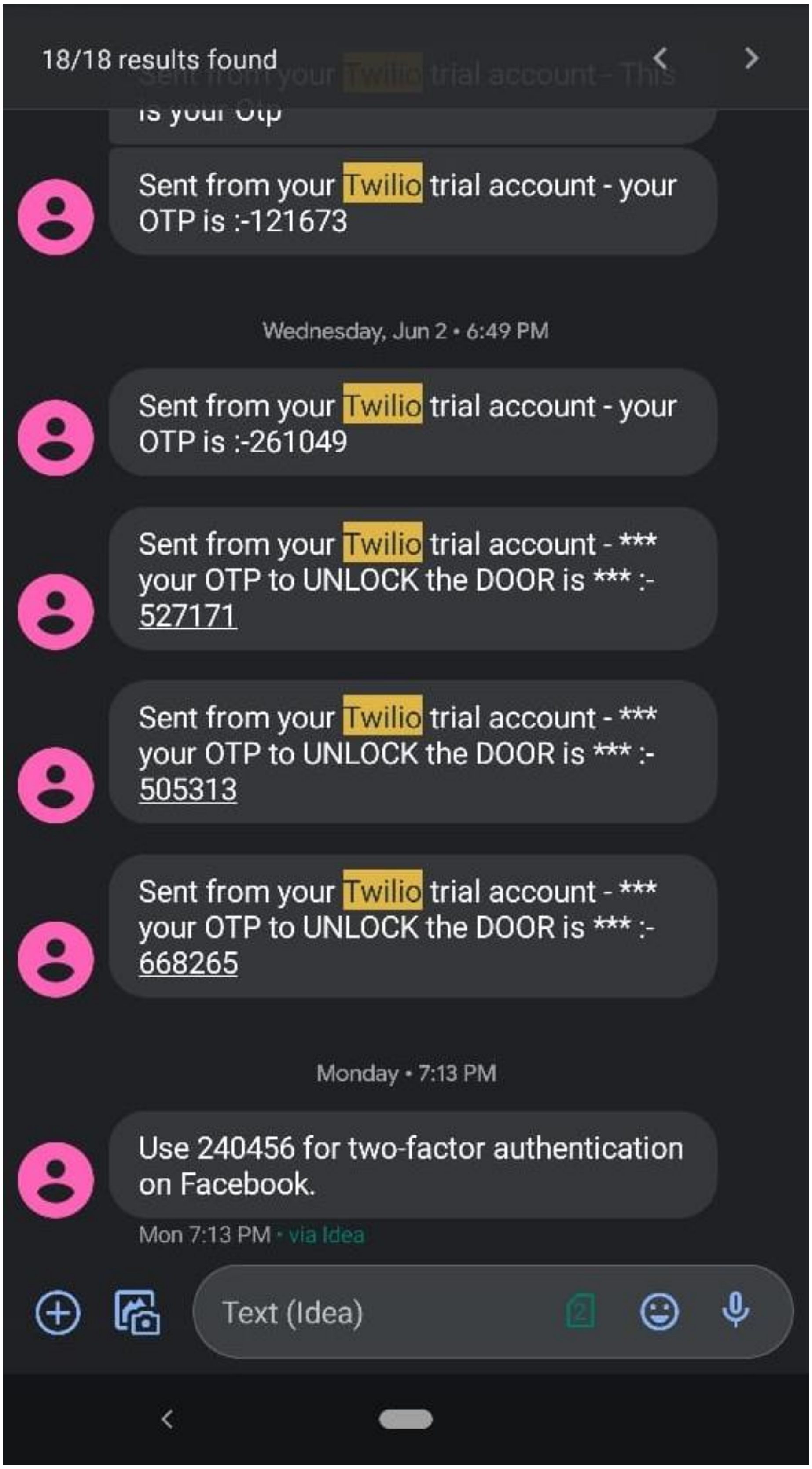
Fig 5.3 GUI WITH OTP OPTION

5.3 ONE TIME PASSWORD (OTP)

A one-time password (OTP) is an automatically generated numeric or alphanumeric string of characters that authenticates the user for a single transaction or login session. An OTP is more secure than a static password, especially a user-created password, which can be weak and/or reused across multiple accounts. OTPs may replace authentication login information or may be used in addition to it in order to add another layer of security.

One-time password examples OTP security tokens are microprocessor-based smart cards or pocket-size key fobs that produce a numeric or alphanumeric code to authenticate access to the system or transaction. This secret code changes every 30 or 60 seconds, depending on how the token is configured. Mobile device apps, such as Google Authenticator, rely on the token device and PIN to generate the one-time password for two-step verification. OTP security tokens can be implemented using hardware, software or on demand. Unlike traditional passwords that remain static or expire every 30 to 60 days, the one-time password is used for one transaction or login session.

This is the one time password which add the extra security to our proposed security system. When the system recognize the face then it will send a OTP to the person registered phone number. And that person have only 3 chances if he/she failed to enter the OTP 3 times then the system will block him/her entry and alert about this failure.



18/18 results found

Sent from your Twilio trial account - This is your Otp

Sent from your Twilio trial account - your OTP is :-121673

Wednesday, Jun 2 • 6:49 PM

Sent from your Twilio trial account - your OTP is :-261049

Sent from your Twilio trial account - *** your OTP to UNLOCK the DOOR is *** :- 527171

Sent from your Twilio trial account - *** your OTP to UNLOCK the DOOR is *** :- 505313

Sent from your Twilio trial account - *** your OTP to UNLOCK the DOOR is *** :- 668265

Monday • 7:13 PM

Use 240456 for two-factor authentication on Facebook.

Mon 7:13 PM • via Idea

Fig 5.3 OTP

When an unauthenticated user attempts to access a system or perform a transaction on a device, an authentication manager on the network server generates a number or shared secret, using one-time password algorithms. The same number and algorithm are used by the security token on the smart card or device to match and validate the one-time password and user. Many companies use Short Message Service (SMS) to provide a temporary passcode via text for a second authentication factor. The temporary passcode is obtained out of band through cell phone communications after the user enters his username and password on networked information systems and transaction-oriented web applications. For two-factor authentication (2FA), the user enters his user ID, traditional password and temporary passcode to access the account or system.

How a one-time password works

In OTP-based authentication methods, the user's OTP app and the authentication server rely on shared secrets. Values for one-time passwords are generated using the Hashed Message Authentication Code (HMAC) algorithm and a moving factor, such as time-based information (TOTP) or an event counter (HOTP). The OTP values have minute or second timestamps for greater security. The one-time password can be delivered to a user through several channels, including an SMS-based text message, an email or a dedicated application on the endpoint. Security professionals have long been concerned that SMS message spoofing and man-in-the-middle (MITM) attacks can be used to break 2FA systems that rely on one-time passwords. However, the U.S. National Institute of Standards and Technology (NIST) announced plans to deprecate the use of SMS for 2FA and one-time passwords, as the method is vulnerable to an assortment of attacks that could compromise those passwords and codes. As a result, enterprises considering deployment of one-time passwords should explore other delivery methods besides SMS.

Benefits of a one-time password

The one-time password avoids common pitfalls that IT administrators and security managers face with password security. They do not have to worry about composition rules, known-bad and weak passwords, sharing of credentials or reuse of the same password on multiple accounts and systems. Another advantage of one-time passwords is that they become invalid in minutes, which prevents attackers from obtaining the secret codes and reusing them.

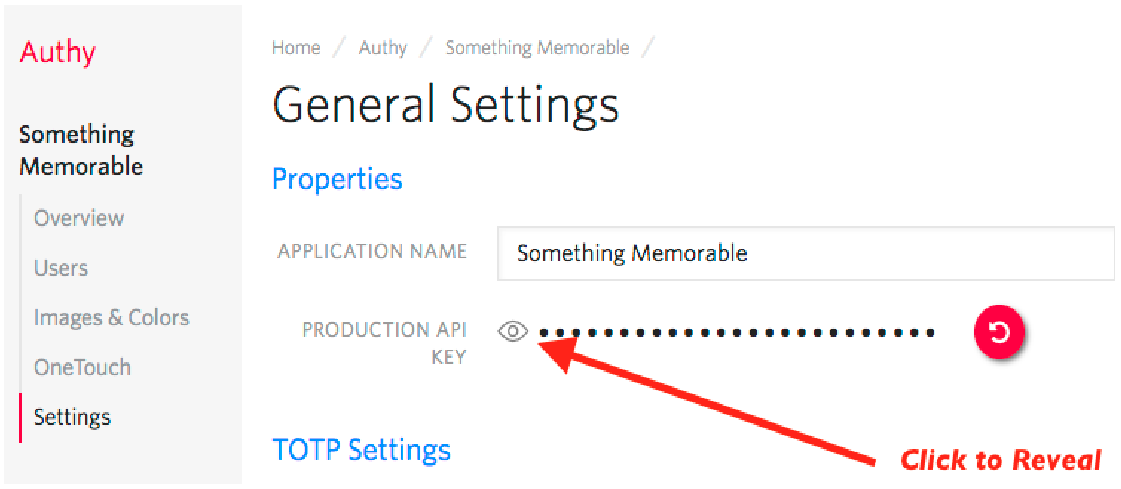
5.3.1 Twilio API (for OTP)

Twilio Messaging is an API to send and receive SMS, MMS, OTT messages globally. It uses intelligent sending features to ensure messages reliably reach end users wherever they are. Twilio has SMS-enabled phone numbers available in more than 180 countries.

How it works:

- User enters their phone number or email
- App generates an authentication token
- App sends the token via selected channel to the user
- User enters the correct token
- App verifies the token

Once a user has been registered with your Twilio Authy application and receives an AuthyID, you can now implement 2FA, passwordless login or protect an in-application high-value transaction. Using the Authy API, you can send one-time passwords over voice or SMS channels. Users can also install the free Authy app or use our SDK to generate offline TOTP codes (soft tokens). Soft tokens do not require wireless connectivity to issue and verify. The Authy API is used to verify a user has access to the right phone number (for SMS and Voice channels) or has access to the right trusted device (for TOTP via the Authy App or use of the SDK). Twilio's Authy API follows the algorithms described in RFC 6238 and RFC 4226 to generate TOTP (Time-Based One-Time Passwords) passwords. It is also possible to use your own hardware tokens, please contact us for information on how to enable this type of 2FA.



Authy

Something Memorable

- Overview
- Users
- Images & Colors
- OneTouch
- Settings

Home / Authy / Something Memorable /

General Settings

Properties

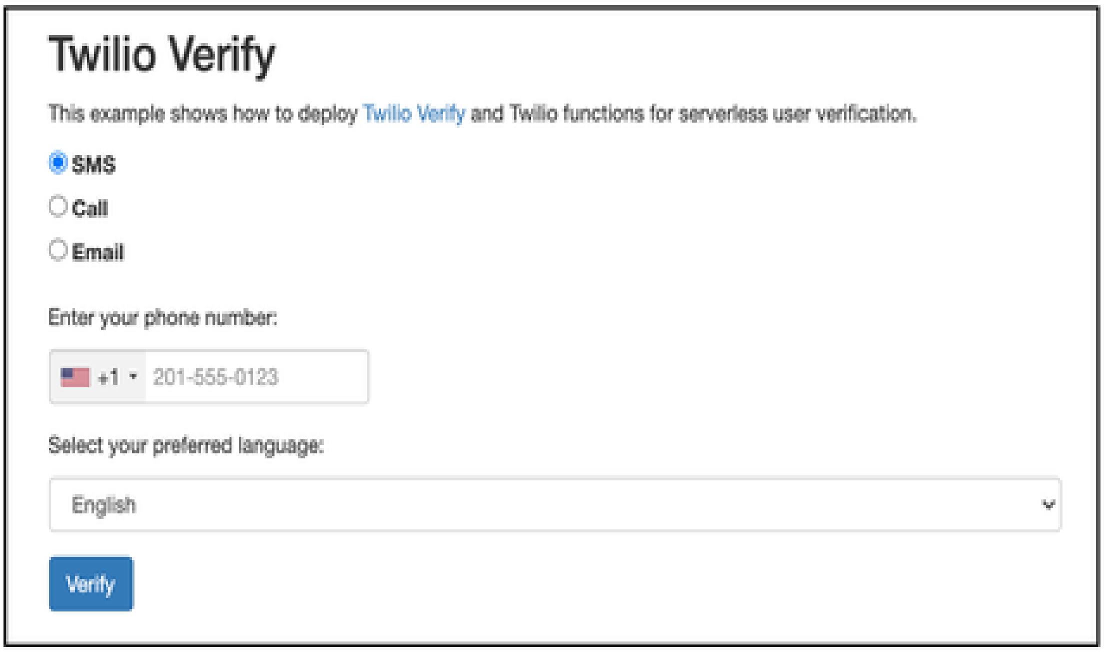
APPLICATION NAME: Something Memorable

PRODUCTION API KEY: ••••••••••••••••••••••••••

TOTP Settings

Fig 5.3.1: TWILIO INTERFACE

Phone Verification made easy with Twilio Verify



Twilio Verify

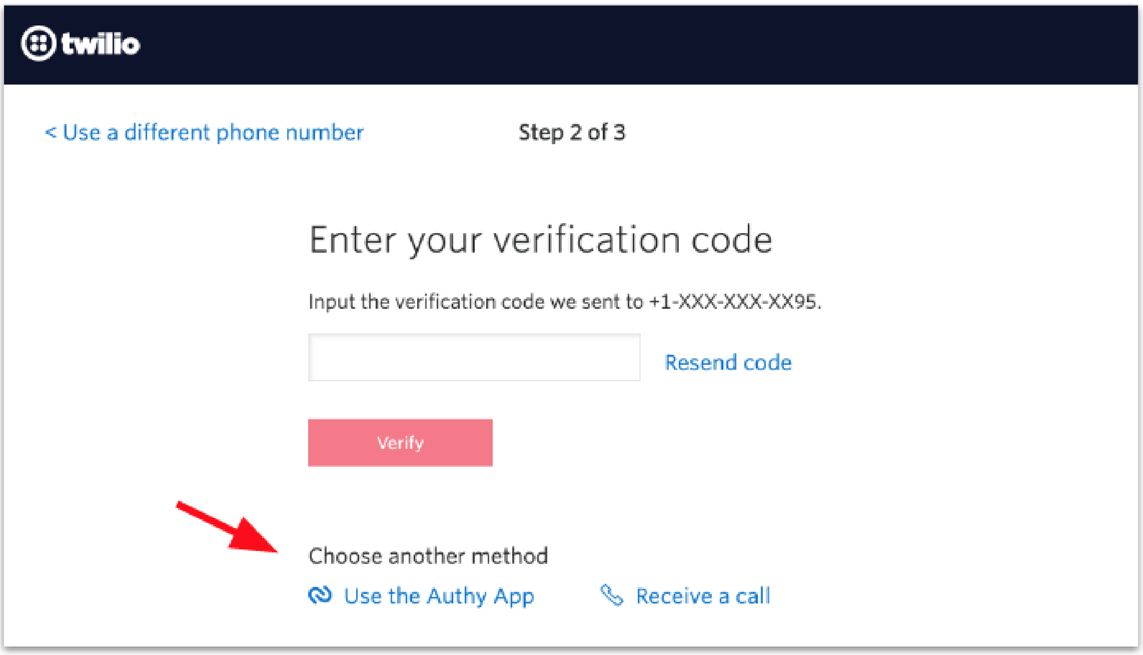
This example shows how to deploy Twilio Verify and Twilio functions for serverless user verification.

- [✓ SMS]
- Call
- Email

Enter your phone number: +1 201-555-0123

Select your preferred language: English

Fig 5.3.2 TWILIO PHONE VERIFICATION


twilio

< Use a different phone number

Step 2 of 3

Enter your verification code

Input the verification code we sent to +1-XXX-XXX-XX95.

Resend code

Verify

Choose another method

Use the Authy App

Receive a call

Fig 5.3.3: TWILIO PHONE VRIFIED

- In order to do phone verification, our app will follow the following workflow:
- The user enters three bits of information to our application
- Phone number
- Country code
- Whether they want to verify by voice or SMS
- Our application makes a request to the Verify API to initiate the verification
- The user receives a call or text message with a 4 digit code
- The user then enters that code into a form in the application
- The application sends the code off to the Verify API, with their phone number and country code again, to verify
- If it is correct then the user has verified their number

CHAPTER 6

DESIGN AND IMPLEMENTATION

6.1 BRIEF

Now-A-days in this world is connected to the smart devices there is a crucial need that is to improve the existing objects and make them smart. Especially to our door locks. To make any object smart enabled we need to explore allits existing drawbacks and add some extra specifications. The major disadvantages in a nor-mal door locking system is that anybody can open a normal door by a duplicate key and it's very difficult if we want our friends and family to get into our house. This is why we cannot decrease these problems. So just change this normal door locking system into a facial recognition enabled smart door lock, which we can open the door whenever we want, So this implementation has come where devices can interact with the users and at the same time verify the safety and keeping them smart. The main concept is that to design and model this Facial recognition door unlock in advanced knowledge of the microcontrollers and interfacing the requirements, as the nodemcu computing is used and been inter-faced with the different devices along with the development of application to develop a GUI application. By a thorough study on the libraries and there functions we has generate the code, we does a most dependable and perfect facial recognition with the new and effective use of the hardware. It is how I initiated and making life easy and interconnected with objects. This will con-tribute a major change in Home Security.

The one time password which add the extra security to our proposed security system. When the system recognize the face then it will send a OTP to the person registered phone number. And that person have only 3 chances if he/she failed to enter the OTP 3 times then the system will block him/her entry and alert about this failure.

6.2 LIMITATIONS

- For OTP we have used Twilio API , It sends only one OTP at a time because we are using trial version that’s why , if we purchased full OTP verification it would be work awesome.
- Real Time Face Recognition algorithm used in this system detect some time unknown person only when faces would be similar to known person.
- The smart facial door unlock system detect a known person from image/video/pictures some times.
- High Dependency on Internet and Mobile Signal: We need a strong signal and good internet connection so that the system should operate in meaningful manner.

6.3 BLCOK DIAGRAM

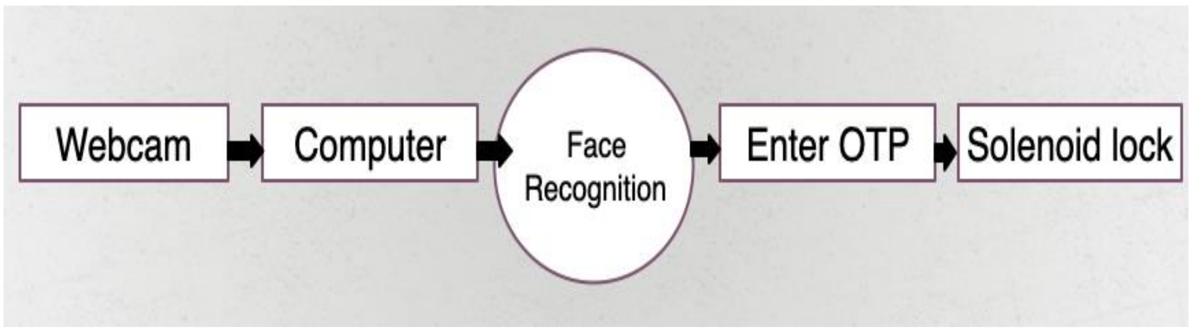


Fig 6.3 BLOCK DIAGRAM

6.4 COMPLETE CIRCUIT DIAGRAM

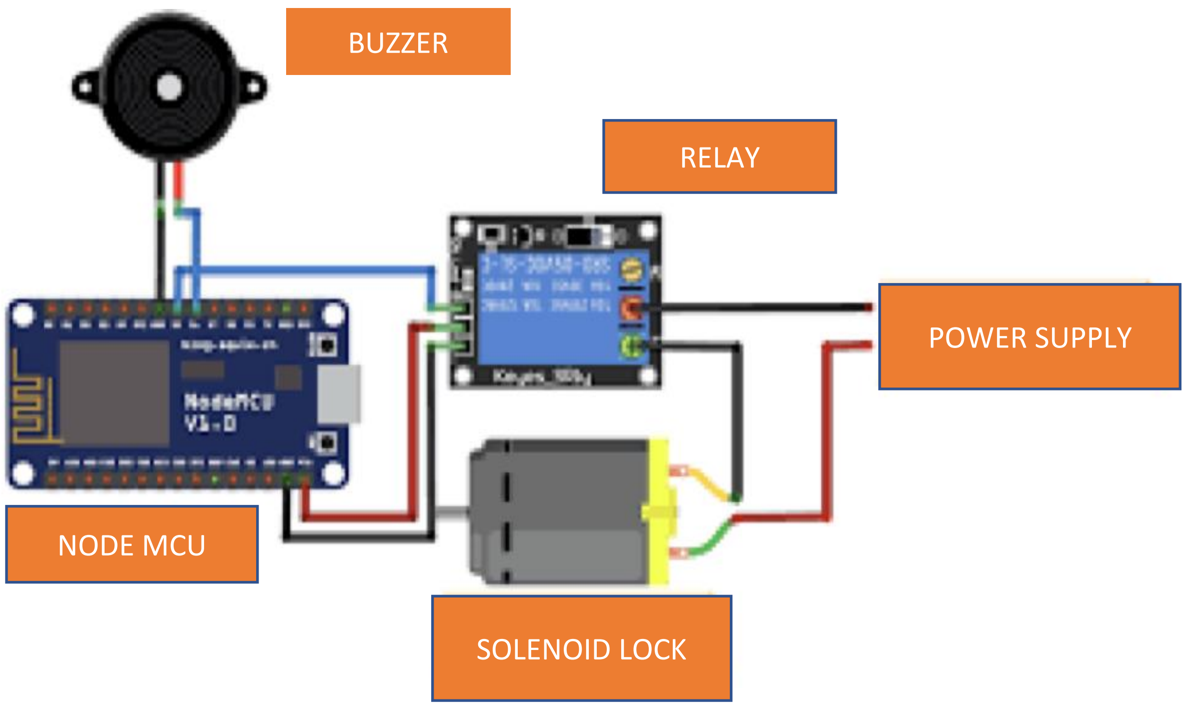


Fig 6.4: CIRCUIT DIAGRAM

6.5 SOURCE CODE

```
# To generate OTP
import random

#To get date and time
import datetime
import time

# Import Module
from tkinter import *


#twilio sms gateway

from twilio.rest import Client

account_sid = 'AC0cebfdb6fced5357cecf78ee1e8xe830'
auth_token = '1aaefcbbb8e32543012ceaffm841c8c1'
client = Client(account_sid, auth_token)

#To connect with microcontroller
import serial
import time
arduino=serial.Serial('COM7',9600)

#To recognize face
```

```
import cv2

import numpy as np

import face_recognition

# This is a running face recognition on live video from your webcam. It's a little
more complicated than the

# other example, but it includes some basic performance tweaks to make things run
a lot faster:

#   1. Process each video frame at 1/4 resolution (though still display it at full
resolution)

#   2. Only detect faces in every other frame of video.


# PLEASE NOTE: This example requires OpenCV (the `cv2` library) to be
installed only to read from your webcam.

# OpenCV is *not* required to use the face_recognition library. It's only required if
you want to run this

# specific. If you have trouble installing it, try any of the other that don't require it
instead.


# Get a reference to webcam #0 (the default one)

video_capture = cv2.VideoCapture(0)


# Load a sample picture and learn how to recognize it.

image1 = face_recognition.load_image_file("image_3.jpeg")

image1_face_encoding = face_recognition.face_encodings(image1)[0]


# Load a second sample picture and learn how to recognize it.

#image2 = face_recognition.load_image_file("image_4.jpeg")
```

```
#image2_face_encoding = face_recognition.face_encodings(image2)[0]


# Create arrays of known face encodings and their names
known_face_encodings = [
    image1_face_encoding
]
known_face_names = [
    "Sumit"
]


# Initialize some variables
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True


e=0


# Import Module
from tkinter import *



while True:
```

```
#choice for user
def choice():
    ch=input("enter N for next or q for exit: ")
    if ch=="N":
        print("Recoginsing Face...")
    elif ch=="q":
        pass
    else:
        print("Wrong Input!")
        choice()

# Grab a single frame of video
ret, frame = video_capture.read()

# Resize frame of video to 1/4 size for faster face recognition processing
small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)

# Convert the image from BGR color (which OpenCV uses) to RGB color
(which face_recognition uses)
rgb_small_frame = small_frame[:, :, ::-1]

# Only process every other frame of video to save time
if process_this_frame:
    # Find all the faces and face encodings in the current frame of video
    face_locations = face_recognition.face_locations(rgb_small_frame)
```

```
        face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations)


        face_names = []
        for face_encoding in face_encodings:
            # See if the face is a match for the known face(s)
            matches = face_recognition.compare_faces(known_face_encodings,
face_encoding)
            name = "Unknown"


            # # If a match was found in known_face_encodings, just use the first one.
            # if True in matches:
            #     first_match_index = matches.index(True)
            #     name = known_face_names[first_match_index]


            # Or instead, use the known face with the smallest distance to the new face
            face_distances = face_recognition.face_distance(known_face_encodings,
face_encoding)
            best_match_index = np.argmin(face_distances)
            if matches[best_match_index]:


                print("Face Recoginsed!")
                name = known_face_names[best_match_index]




                face_names.append(name)
        process_this_frame = not process_this_frame


        # Display the results
        for (top, right, bottom, left), name in zip(face_locations, face_names):
            # Scale back up face locations since the frame we detected in was
scaled to 1/4 size
            top *= 4
            right *= 4
            bottom *= 4
            left *= 4


            # Draw a box around the face
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)


            # Draw a label with a name below the face
            cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255),
cv2.FILLED)
            font = cv2.FONT_HERSHEY_DUPLEX
            cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255,
255), 1)


        # Display the resulting image
        cv2.imshow('Video', frame)


        #Getting date and time
```

```
x=str(datetime.datetime.now())

y=x.replace(':','-')[:-7]


# writing the extracted images

s_name='DATA/'+y+' '+name+'.jpg'

cv2.imwrite(s_name, frame)


#generating random OTP

otp=random.randint(100000,999999)

#print(otp)


# OTP will be send after face is recognised

message = client.messages \

.create(

    body='*** your OTP to UNLOCK the DOOR is *** :- '+str(otp),

     from_='+19163786591',

     to='+917599208825'

 )



# create root window

root = Tk()


# root window title and dimension

root.title("Smart Facial Door Unlock")
```

```
# Set geometry (widthxheight)
root.geometry('600x500')

root.config(background = "black", pady=10)



get_OTP=StringVar()
attempt=0
def button():
    global attempt
    attempt+=1
    global u_otp
    global upw
    u_otp=get_OTP.get()
    #three attempts are allowed for OTP verification
    #otp_v=input("please enter OTP: ")
    # verify the OTP
    if u_otp==str(otp):
        print("Authentication Success!")
        print("Welcome",name)
        print("door is UNLOCKED!\n")

        heading= Label(root, font=( 'aria' ,40, 'bold' ),text="DOOR
UNLOCKED..!!",
                       fg="orange",bd=10,bg='black')
```

```
          heading.grid(row=3,column=1)




          arduino.write(b'1')

          print("You have 5 seconds to enter the room")


          for i in range(1,6):

              print(i,"...")

              time.sleep(1)

          print("Door is LOCKED again!")

          arduino.write(b'0')

          root.destroy()

          choice()



      else:

          if attempt<3:

              print("attempt",attempt,"Authentication Error!")

              heading= Label(root, font=( 'aria' ,40, 'bold' ),text="Attempt
Failed",

                         fg="orange",bd=10,bg='black')

              heading.grid(row=7,column=1)


          else:
              print("All attempts are failed...")

              print("Unknown Access!")

              print("Access Denied!")

              print(" BUZZER<<BEEP>><<BEEP>>\n")



              heading= Label(root, font=( 'aria' ,40, 'bold' ),text="UNKNOWN
ACCESS",

                         fg="orange",bd=10,bg='black')

              heading.grid(row=7,column=1)


              arduino.write(b'2')


              for i in range(1,10):


                time.sleep(1)


              #arduino.write(b'0')

              root.destroy()

              choice()


        def next():

      root.destroy()

      choice()
```

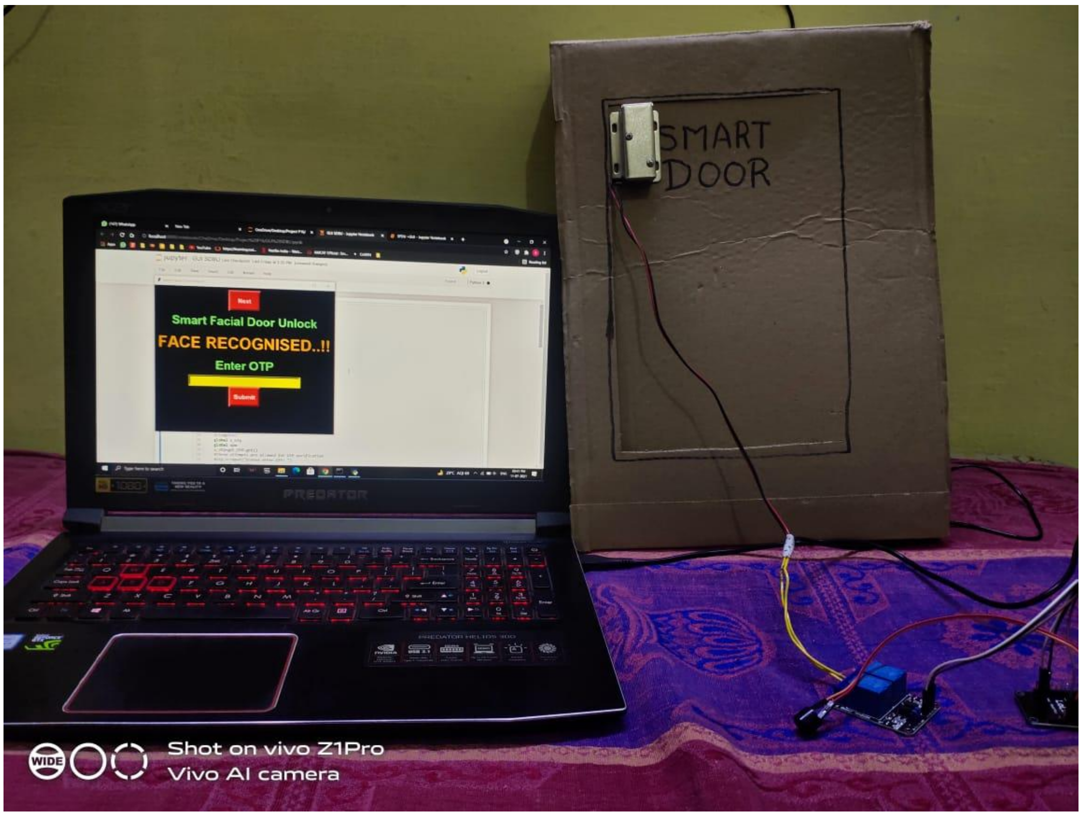
```
        # all widgets will be here

        heading= Label(root, font=( 'aria' ,30, 'bold' ),text="Smart Facial Door
Unlock",

                  fg="lightgreen",bd=10,bg='black')

        heading.grid(row=1,column=1)



        heading= Label(root, font=( 'aria' ,40, 'bold' ),text="FACE
RECOGNISED..!!",

                  fg="orange",bd=10,bg='black')

        heading.grid(row=2,column=1)



        b=Button(root,padx=16,pady=8, bd=10 ,fg="white",

                 font=('ariel' ,16,'bold'),width=4,

                 text="Submit", bg="red", command=button)

        b.grid(row=6,column=1)



        b=Button(root,padx=16,pady=8, bd=10 ,fg="white",

                 font=('ariel' ,16,'bold'),width=4,

                 text="Next", bg="red", command=next)

        b.grid(row=0,column=1)
```

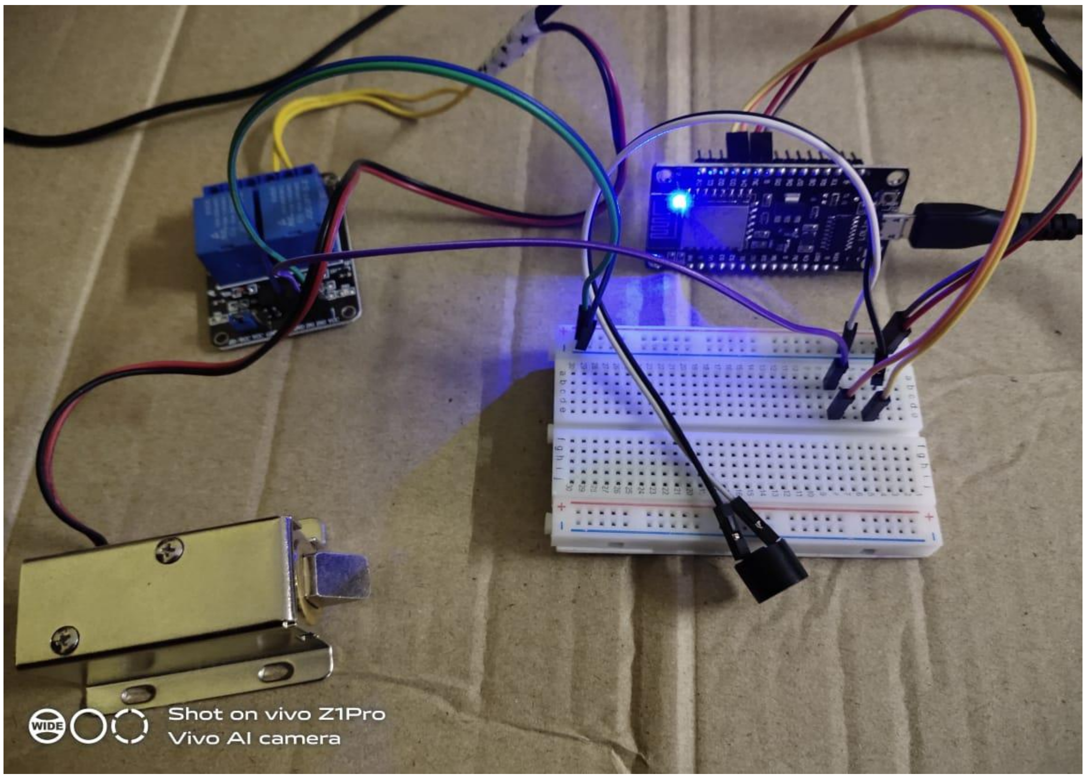
```
            OTP_e= Entry(root,font=('ariel' ,16,'bold'),
                         bd=6,insertwidth=4,
                         bg="yellow" ,justify='left',width=30, textvariable=get_OTP)
            OTP_e.grid(row=5,column=1)



            OTP_l= Label(root, font=( 'aria' ,30, 'bold' ),text="Enter OTP",
                         fg="lightgreen",bd=10,bg='black')
            OTP_l.grid(row=4,column=1)



            # Execute Tkinter
            root.mainloop()
         else:
            face_names.append(name)

            process_this_frame = not process_this_frame

            # Display the results
            for (top, right, bottom, left), name in zip(face_locations, face_names):
               # Scale back up face locations since the frame we detected in was
scaled to 1/4 size
               top *= 4
               right *= 4
               bottom *= 4
               left *= 4
            # Draw a box around the face
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

            # Draw a label with a name below the face
            cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255),
cv2.FILLED)
            font = cv2.FONT_HERSHEY_DUPLEX
            cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255,
255), 1)



          #Getting date and time
          x=str(datetime.datetime.now())
          y=x.replace(':','-')[:-7]

          # writing the extracted images
          s_name='DATA/'+y+' '+name+'.jpg'
          cv2.imwrite(s_name, frame)



          # Display the resulting image
          cv2.imshow('Video', frame)
```

```
                print("Unknown Access!")
                print("Access Denied!")
                print(" BUZZER<<BEEP>><<BEEP>>\n")



                arduino.write(b'2')

                for i in range(1,10):

                    time.sleep(1)

                arduino.write(b'0')

                choice()
            face_names.append(name)



    process_this_frame = not process_this_frame

    # Display the results
    for (top, right, bottom, left), name in zip(face_locations, face_names):
        # Scale back up face locations since the frame we detected in was scaled to 1/4
size
        top *= 4
        right *= 4
        bottom *= 4
        left *= 4

        # Draw a box around the face
        cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

        # Draw a label with a name below the face
        cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255),
cv2.FILLED)
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)

    # Display the resulting image
    cv2.imshow('Video', frame)

    # Hit 'q' on the keyboard to quit!
    if cv2.waitKey(1) & 0xFF == ord('q'):
        print("Thank You! Successfully Exit...")
        break

# Release handle to the webcam
video_capture.release()
cv2.destroyAllWindows()
```

6.6 SNAPSHOTS

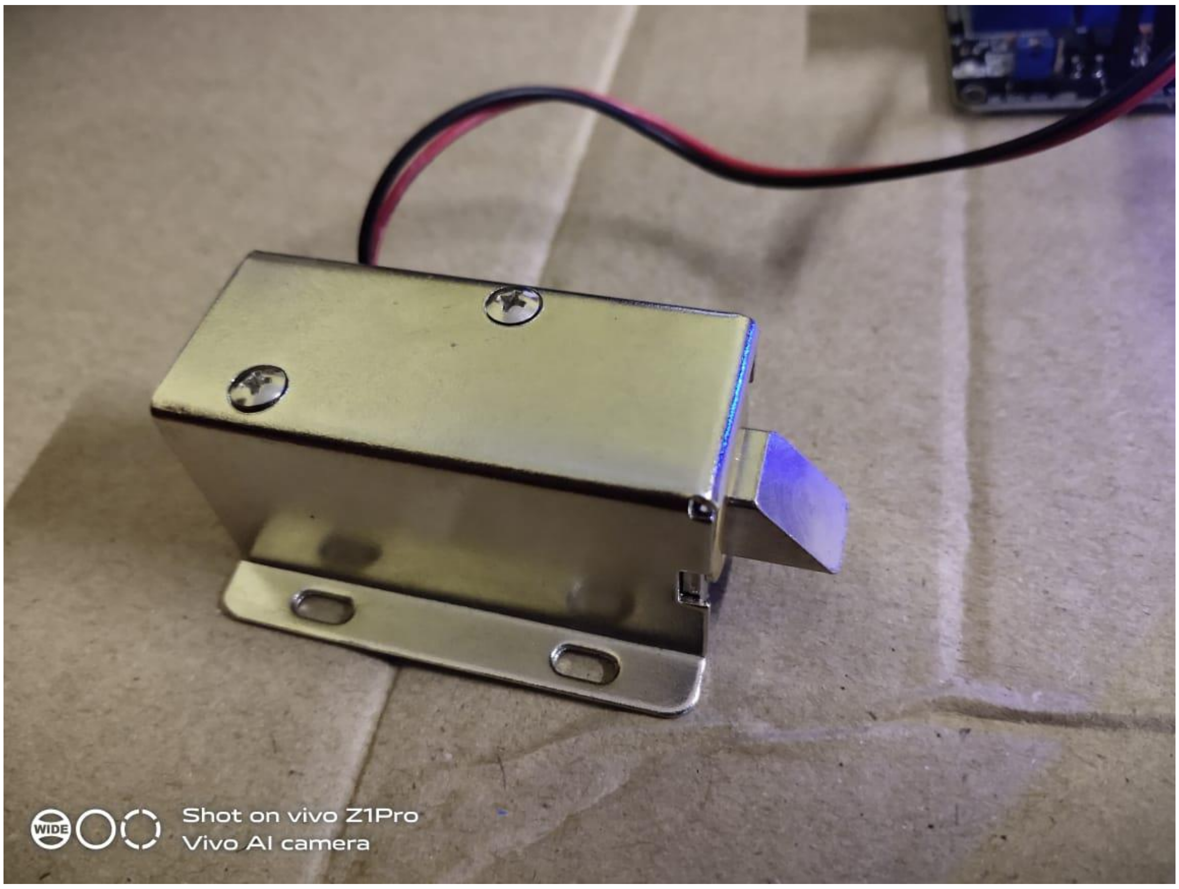
Snapshot -1



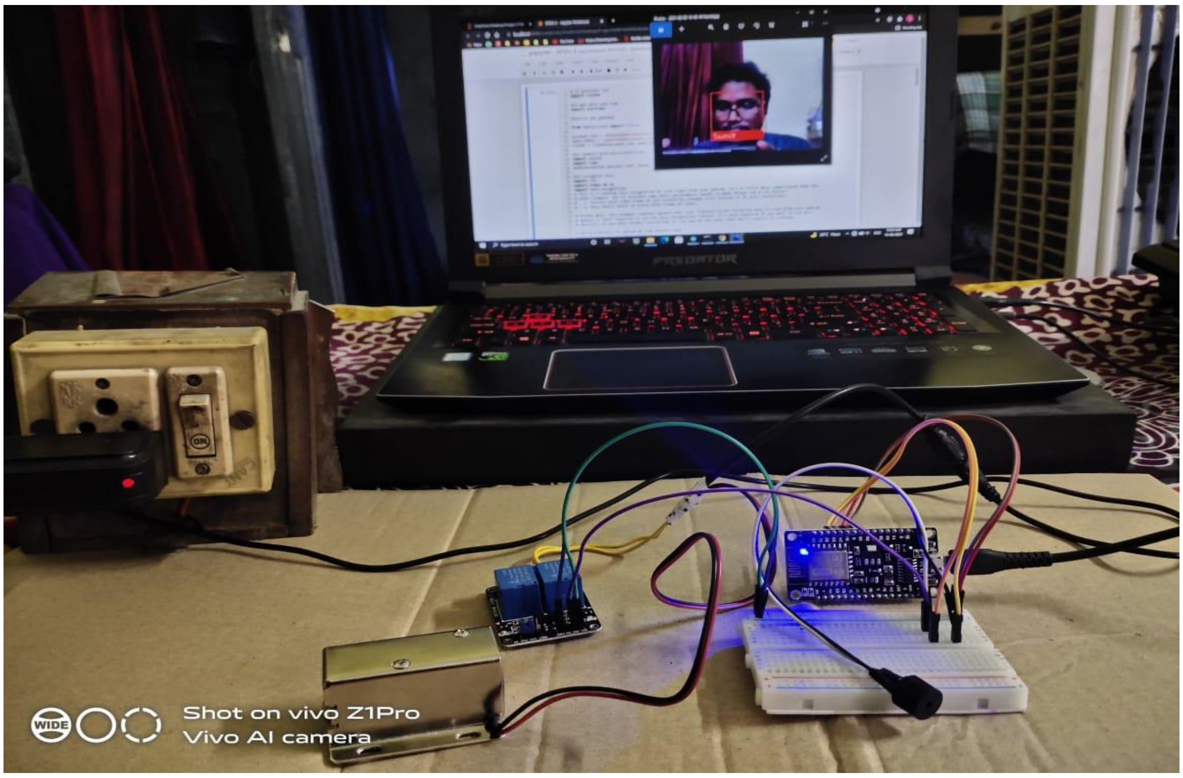


Snapshot-2

Snapshot-3



Snapshot-4



CHAPTER 7

CONCLUSION

Face recognition based door locking has been developed to provide better security. It is user friendly system. The use of Eigen face recognition technique makes system more secure. This system can be used in several places where high security is required where confidential information and equipment is kept. For example, research institutes, banks, forensic Laboratories. This system can also be used for domestic purposes. This project helps to reduce problem of thefts and frauds. In case of unauthorized person's entry, system alerts authorized person with SMS and at the same time the buzzer beeps to alert people. This is a cost efficient and reliable door locking system. This system can be used in many places where need of security is maximum and security cannot be compromised.

In this system we have implemented the Smart Facial Door Unlock system using Face Recognition (Image Processing) NodeMCU, Solenoid Lock, Relay, Python, Open CV and Twilio API (For OTP). The system in able to accurately detect and recognize the face and after that user get OTP, if the user put correct OTP and matched with system the door is unlock otherwise user will unable to enter in the room/office and the system will give alert and beep the buzzer. User will get three chances to put correct OTP if the user fails system will give alert and beep the buzzer. The Implemented system is moderated cost, so that it is an affordable to the everyone.

REFERENCES

[1] Door Unlock using Face Recognition (Abdul Azeem, Sathuluri Mallikarjun Rao,Kandula Rama Rao, Shaik Akbar Basha, Harsha Pedarla, Modela Gopi)

[2] Makers Pro Projects(Simple Door Lock Using Face recognition Solenoid Lock)(Muhammad Aqib)

[3]. JayantDabhade, AmirushJavare, TusharGhayal, AnkurShelar, "Smart Door Lock System: Improving Home Security using Bluetooth Technology" , International Journal of Computer Applications (0975 – 8887) Volume 160 –No 8, February 2017

[4]. Md. Nasimuzzaman Chowdhury, Md. ShibleeNooman, SrijonSarker, "Access Control of Door and Home Security by Raspberry Pi through Internet" , International Journal of Scientific & Engineering Research, Volume 4, Issue11, November-2013 ISSN 2229-5518

[5]. Sanchita Bilgaiyan, Sherin James, Sneha. S Bhonsle, Shruti Shahdeo, Keshavamurthy,‖ Android Based Signboard Detection using Image and Voice Alert System‖ IEEE International Conference, May 20-21, 2016, India

[6] Dhiraj Sunehra , Ayesha Bano, ―An Intelligent Surveillance with Cloud Storage for Home Security‖ , Annual IEEE India Conference (INDICON), 2014.

[7] S.Padmapriya & Esther Annlin KalaJames, ―Real Time Smart Car Lock

[8] (2016) "Arduino Based Door Unlocking System with RealTime Control" Somjit Nath, Paramita Banerjee, Rathindra Nath, Biswas, Swarup Kumar, Mitra

[9] (2014) K.Gopalakrishnan, V.Sathish Kumar "embedded image capturing system using the raspberry pi system" international Journal.

[10] (2014) “Development of Intelligent Automatic Door System” Daiki Nishida, Kumiko Tsuzura1, Shunsuke Kudoh1, Kazuo Takai, Tatsuhiro Momodori.

[11](2012) “Face Recognition Based on Magnetic Door Lock System Using Microcontroller” Harnani Hassan, Raudah, Abu Bakar Ahmad Faculty of Electrical Engineering.

[12] Twilio API for OTP.