PROJECT APPROVAL OF

**“HOME SECURITY USING FACE RECOGNITION TECHNOLOGY”**

A Group Project Report

Submitted in Partial Fulfillment of the Requirements

For the award of the Degree of

**Bachelor of Technology in**

**Electrical and Computer Engineering (ECM)**

By

**G.Sharan 17311A19f1**

**V.sai.madhav 17311A19h0**

**G.Shripadh rao 17311A19h6**

#### Approved by:

Internal Guide: Mr.Nanda Kumar Project Coordinator:Mrs.Naga Shailaja

HOD: Dr.K.Sasidhar

****

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING (EEE)

**SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY**

November 2019

****

**SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY**

A Group Project Report

Submitted in Partial Fulfillment of the Requirements

For the award of the Degree of

**Bachelor of Technology in**

**Electrical and Computer Engineering (ECM)**

By

**G.sharan 17311A19f1**

**V.sai.madhav 17311A19h0**

**G.shripadh rao 17311A19h6**

Under the Guidance / Supervision of

#### Mr.Nanda kumar

**Department of Electrical and Computer Engineering (ECM)**

November 2019

**SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(AUTONOMOUS)**

****

**DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING (ECM)**

**CERTIFICATE**

This is to certify that the Group Project work titled **“HOME SECURITY USING FACE RECOGNITION TECHNOLOGY”**, submitted by **G.Sharan** bearing **Roll No. 17311A19F1**,**V.Sai Madhav** bearing **Roll No. 17311A19H0** and **G.Shripadh Rao** bearing **Roll No. 17311A19H6** towards partial fulfillment for the award of Bachelors Degree in Electrical and Computer Engineering (ECM) from Sreenidhi Institute of Science & Technology, Ghatkesar, Hyderabad, is a record of bonafide work done by him/ her. The results embodied in the work are not submitted to any other University or Institute for award of any degree or diploma.

#### Mr. Nanda Kumar

**Dr.K.Sasidhar**

Assistant.Professor,ECM Professor & HOD,ECM

**External Examiner**

**ACKNOWLEDGEMENT**

We would like to thank my Supervisor **Mr.Nanda Kumar,** Assistant professor of ECM and our group project coordinator **Mrs.Naga Shailaja,** Assistant professor of ECMfor giving me their constant guidance, support and motivation throughout the period this course work was carried out. His readiness for consultation at all times, his educative comments and assistance even with practical things have been invaluable. I am thankful that he gave me the freedom to do the work with my ideas.

We express my sincere gratitude to **Dr. K. Sasidhar**, Head of Department of ECM for helping me in carrying out this project giving support throughout the period of my study in SNIST.

# Finally, we extend our sense of gratitude to almighty, our parents, teaching and non teaching staff who helped us in this Endeavour

1. **Sharan**
2. **(Roll No:17311A19F1)**

**V.Sai Madhav**

**(Roll No:17311A19H0)**

1. **Shripadh Rao**
2. **(Roll No:17311A19H6)**

**ABSTRACT**

**AIM:** To make a home security system using face recognition technology.

**COMPONENTS:** 

Nodemcu(esp8266)

 Servo motor

 Bread Board

 Connecting wires

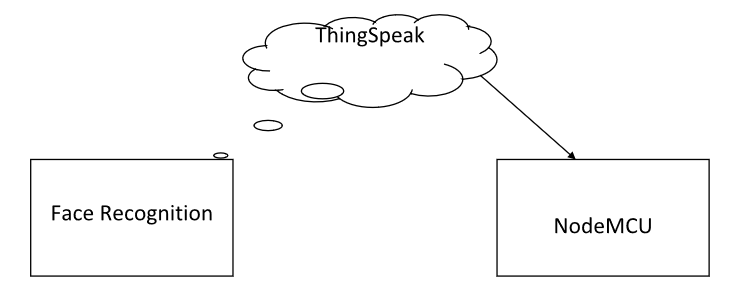
**SOFTWARES USED:**

**** Python 3.7.4

 Cloud – Thing Speak

 Arduino 1.8.9

**BLOCK DIAGRAM:**

****

**DESCRIPTION:**

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity. Human face recognition procedure basically consists of two phases, namely face detection, where this process takes place very rapidly in humans, except under conditions where the object is located at a short distance away, the next is the introduction, which recognize a face as individuals. Stage is then replicated and developed as a model for facial image recognition (face recognition) is one of the much-studied bio metrics technology and developed by experts. There are two kinds of methods that are currently popular in developed face recognition pattern namely, Eigen face method and Fisher-face method. Facial image recognition Eigen face method is based on the reduction of face- dimensional space using Principal Component Analysis (PCA) for facial features. The main purpose of the use of PCA on face recognition using Eigen faces was formed (face space) by finding the eigen vector corresponding to the largest eigenvalue of the face image. The area of this project face detection system with face recognition is Image processing. The software requirements for this project is mat lab software.

**ADVANTAGES:**

* Increased Security: One of the biggest pros of facial recognition technology is that it enhances safety and security.
* Fast and Accurate: With the ever-increasing demand for speed and the growing number of cyber attacks, having fast and accurate technology is key.

**DISADVANTAGES:**

* Data Storage : The video and high-quality images required for facial recognition take up a significant amount of storage.The accuracy of word recognition reduces in face of the noise.
* Changes in Appearance and Camera Angle: Any major changes in appearance, including facial hair and weight changes, can throw off the technology

**FUTURE SCOPE:**

Today, one of the fields that use facial recognition the most is security. Facial recognition is very effective tools that can help law enforcers recognize criminals and software companies are leveraging the technology to help users access their technology. This technology can be further developed to be used in other avenues such as ATMs, accessing confidential files, or other sensitive materials. This can make other security measures such as passwords and keys obsolete.

**CONTENTS**

1. **INTRODUCTION**
   1. **Face Recognition**
   2. **Face Detection**
   3. **Neural Network**
   4. **Bluetooth Communication**
2. **TECHNICAL OBJECTIVES**

**2.1 Speech Recognition**

**2.2 Arduino and Bluetooth**

**2.3 Wireless Communication**

**3. COMPONENTS**

**3.1 Node MCU**

**3.2 Servo Motor**

**3.3 d-lib**

**3.4 ThingSpeak**

**4. RESULTS**

**LIST OF FIGURES**

**CHAPTER-1**

**INTRODUCTION**

Face recognition is the task of identifying an already detected object as a known or unknown face. Often the problem of face recognition is confused with the problem of face detection Face Recognition on the other hand is to decide if the "face" is someone known, or unknown, using for this purpose a database of faces in order to validate this input face

* 1. **FACE RECOGNITION :**

DIFFERENT APPROACHES OF FACE RECOGNITION:

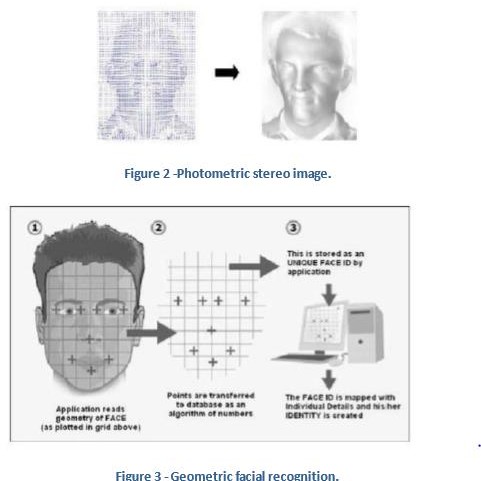
There are two predominant approaches to the face recognition problem: Geometric (feature based) and photo metric (view based). As researcher interest in face recognition continued, many different algorithms were developed, three of which have been well studied in face recognition literature.

### Recognition algorithms can be divided into two main approaches:

* + 1. **Geometric:** Is based on geometrical relationship between facial landmarks, or in other words the spatial configuration of facial features. That means that the main geometrical features of the face such as the eyes, nose and mouth are first located and then faces are classified on the basis of various geometrical distances and angles between features.
    2. **Photo metric stereo:** Used to recover the shape of an object from a number of images taken under different lighting conditions. The shape of the recovered object is defined by a gradient map, which is made up of an array of surface normal (Zhao and Chellappa, 2006)

### Popular recognition algorithms include:

* + - 1. Principal Component Analysis using Eigen faces, (PCA)
      2. Linear Discriminate Analysis,
      3. Elastic Bunch Graph Matching using the Fisher-face algorithm
      4. Face\_Recognition (Using d lib)



## 

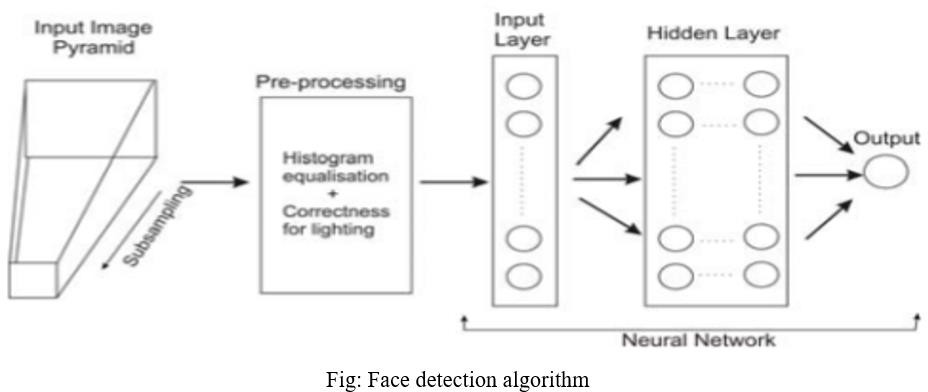
## FACE DETECTION:

Face detection involves separating image windows into two classes; one containing faces taring the background (clutter). It is difficult because although commonalities exist between faces, they can vary considerably in terms of age, skin colour and facial expression. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down into two steps.

The first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image. The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height).

### The face detection system can be divided into the following steps :-

1. **Pre-Processing:** To reduce 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.
2. **Classification:** Neural networks are implemented to classify the images as faces or non faces by training on these examples. We use both our implementation of the neural network provided by the d-lib for this task. Different network configurations are experimented with to optimize the results.
3. **Localization:** The trained neural network is then used to search for faces in an image and if present localize them in a bounding box. Various Feature of Face on which the work has done on:- Position Scale Orientation Illumination



* 1. **Neural Network:**

Neural networks gaining much more attention in many pattern recognition problems, such as OCR, object recognition, and autonomous robot driving. Since face detection can be treated as a two class pattern recognition problem, various neural network algorithms have been proposed.

**The advantage of using neural networks for** **face detection is:**

the feasibility of training a system to capture the complex class conditional density of face patterns. However, one demerit is that the network architecture has to be extensively tuned (number of layers, number of nodes, learning rates, etc.) to get exceptional performance. In early days most hierarchical neural network was proposed by Agui et al. The first stage having two parallel sub networks in which the inputs are filtered intensity values from an original image. The inputs to the second stage network consist of the outputs from the sub networks and extracted feature values. An output at the second stage shows the presence of a face in the input region. Propp and Samal developed one of the earliest neural networks for face detection. Their network consists of four layers with 1,024 input units, 256 units in the first hidden layer, eight units in the second hidden layer, and two output units. Feraud and Bernier presented a detection method using auto associative neural networks. The idea is based on which shows an auto associative network with five layers is able to perform a nonlinear principal component analysis. One auto associative network is used to detect frontal- view faces and another one is used to detect faces turned up to 60 degrees to the left and right of the frontal view. After that Lin et al. presented a face detection system using probabilistic decision-based neural network (PDBNN). The architecture of PDBNN is similar to a radial basis function (RBF) network with modified learning rules and probabilistic interpretation.

**Significant commands used:**

1. **face\_recognition.api.batch\_face\_locations(*images*, *number\_of\_times\_to\_upsample=1*, *batch\_size=128*)**

Returns an 2d array of bounding boxes of human faces in a image using the CNN face detector If you are using a GPU, this can give you much faster results since the GPU can process batches of images at once. If you aren’t using a GPU, you don’t need this function.

|  |  |
| --- | --- |
| **Parameters:** | * **img** – A list of images (each as a numpy array) * **number\_of\_times\_to\_upsample** – How many times to upsample the image looking for faces. Higher numbers find smaller faces. * **batch\_size** – How many images to include in each GPU processing batch. |
| **Returns:** | A list of tuples of found face locations in css (top, right, bottom, left) order |

1. **face\_recognition.api.compare\_faces(*known\_face\_encodings*, *face\_encoding\_to\_check*, *tolerance=0.6*)**

Compare a list of face encodings against a candidate encoding to see if they match.

|  |  |
| --- | --- |
| **Parameters:** | * **known\_face\_encodings** – A list of known face encodings * **face\_encoding\_to\_check** – A single face encoding to compare against the list * **tolerance** – How much distance between faces to consider it a match. Lower is more strict. 0.6 is typical best performance. |
| **Returns:** | A list of True/False values indicating which known\_face\_encodings match the face encoding to check |

1. **face\_recognition.api.face\_distance(*face\_encodings*, *face\_to\_compare*)**

Given a list of face encodings, compare them to a known face encoding and get a euclidean distance for each comparison face. The distance tells you how similar the faces are.

|  |  |
| --- | --- |
| **Parameters:** | * **faces** – List of face encodings to compare * **face\_to\_compare** – A face encoding to compare against |
| **Returns:** | A numpy ndarray with the distance for each face in the same order as the ‘faces’ array |

1. **face\_recognition.api.face\_encodings(*face\_image*, *known\_face\_locations=None*, *num\_jitters=1*)**

Given an image, return the 128-dimension face encoding for each face in the image.

|  |  |
| --- | --- |
| **Parameters:** | * **face\_image** – The image that contains one or more faces * **known\_face\_locations** – Optional - the bounding boxes of each face if you already know them. * **num\_jitters** – How many times to re-sample the face when calculating encoding. Higher is more accurate, but slower (i.e. 100 is 100x slower) |
| **Returns:** | A list of 128-dimensional face encodings (one for each face in the image) |

1. **face\_recognition.api.face\_landmarks(*face\_image*, *face\_locations=None*, *model='large'*)**

Given an image, returns a dict of face feature locations (eyes, nose, etc) for each face in the image

|  |  |
| --- | --- |
| **Parameters:** | * **face\_image** – image to search * **face\_locations** – Optionally provide a list of face locations to check. * **model** – Optional - which model to use. “large” (default) or “small” which only returns 5 points but is faster. |

1. **face\_recognition.api.face\_locations(*img*, *number\_of\_times\_to\_upsample=1*, *model='hog'*)**

Returns an array of bounding boxes of human faces in a image

|  |  |
| --- | --- |
| **Parameters:** | * **img** – An image (as a numpy array) * **number\_of\_times\_to\_upsample** – How many times to upsample the image looking for faces. Higher numbers find smaller faces. * **model** – Which face detection model to use. “hog” is less accurate but faster on CPUs. “cnn” is a more accurate deep-learning model which is GPU/CUDA accelerated (if available). The default is “hog”. |
| **Returns:** | A list of tuples of found face locations in css (top, right, bottom, left) order |

1. **face\_recognition.api.load\_image\_file(*file*, *mode='RGB'*)**

Loads an image file (.jpg, .png, etc) into a numpy array

|  |  |
| --- | --- |
| **Parameters:** | * **file** – image file name or file object to load * **mode** – format to convert the image to. Only ‘RGB’ (8-bit RGB, 3 channels) and ‘L’ ( black and white ) are supported. |
| **Returns:** | image contents as numpy array |
|  |  |

**CHAPTER-3**

**COMPONENTS**

**3.1 NodeMCU:**

It is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It 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. As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE". This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.



**3.2 Servo motor:**

A servo motor is an electrical device which can push or rotate an object with great precision. If you want to rotate and object at some specific angles or distance, then you use servo motor. It is just made up of simple motor which run through servo mechanism. If motor is used is DC powered then it is called DC servo motor, and if it is AC powered motor then it is called AC servo motor. We can get a very high torque servo motor in a small and light weight packages.Due to these features they are being used in many applications like toy car, RC helicopters and planes, Robotics, Machines etc..

Servo motors are rated in kg/cm (kilogram per centimeter) most hobby servo motors are rated at 3kg/cm or 6kg/cm or 12kg/cm. This kg/cm tells you how much weight your servo motor can lift at a particular distance For example: A 6kg/cm Servo motor should be able to lift 6kg if the load is suspended 1cm away from the motors shaft, the greater the distance the lesser the weight carrying capacity.

The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.



**3.3 d-lib:**

Dlib is a general purpose [cross-platform](https://en.wikipedia.org/wiki/Cross-platform" \o "Cross-platform) software [library](https://en.wikipedia.org/wiki/Library_(computing)" \o "Library (computing)) written in the programming language [C++](https://en.wikipedia.org/wiki/C++" \o "C++). Its design is heavily influenced by ideas from [design by contract](https://en.wikipedia.org/wiki/Design_by_contract" \o "Design by contract) and [component-based software engineering](https://en.wikipedia.org/wiki/Component-based_software_engineering" \o "Component-based software engineering). Thus it is, first and foremost, a set of independent software components. It is [open-source software](https://en.wikipedia.org/wiki/Open-source_software" \o "Open-source software) released under a [Boost Software License](https://en.wikipedia.org/wiki/Boost_(C++_libraries)" \l "License" \o "Boost (C++ libraries)).

Since development began in 2002, Dlib has grown to include a wide variety of tools. As of 2016, it contains software components for dealing with [networking](https://en.wikipedia.org/wiki/Computer_network" \o "Computer network), [threads](https://en.wikipedia.org/wiki/Thread_(computing)" \o "Thread (computing)), [graphical user interfaces](https://en.wikipedia.org/wiki/Graphical_user_interface" \o "Graphical user interface), [data structures](https://en.wikipedia.org/wiki/Data_structure" \o "Data structure), [linear algebra](https://en.wikipedia.org/wiki/Linear_algebra" \o "Linear algebra), [machine learning](https://en.wikipedia.org/wiki/Machine_learning" \o "Machine learning), [image processing](https://en.wikipedia.org/wiki/Image_processing" \o "Image processing), [data mining](https://en.wikipedia.org/wiki/Data_mining" \o "Data mining), [XML](https://en.wikipedia.org/wiki/XML" \o "XML) and text parsing, [numerical optimization](https://en.wikipedia.org/wiki/Numerical_optimization" \o "Numerical optimization), [Bayesian networks](https://en.wikipedia.org/wiki/Bayesian_network" \o "Bayesian network), and many other tasks. In recent years, much of the development has been focused on creating a broad set of statistical machine learning tools and in 2009 Dlib was published in the [Journal of Machine Learning Research](https://en.wikipedia.org/wiki/Journal_of_Machine_Learning_Research" \o "Journal of Machine Learning Research). Since then it has been used in a wide range of domain.

**3.4 ThingSpeak:**

ThingSpeak is an [open-source](https://en.wikipedia.org/wiki/Open-source_software" \o "Open-source software) [Internet of Things](https://en.wikipedia.org/wiki/Internet_of_Things" \o "Internet of Things) (IoT) application and [API](https://en.wikipedia.org/wiki/API" \o "API) to store and retrieve data from things using the [HTTP](https://en.wikipedia.org/wiki/HTTP" \o "HTTP) and MQTT protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates.

ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications.

ThingSpeak has integrated support from the numerical computing software [MATLAB](https://en.wikipedia.org/wiki/MATLAB" \o "MATLAB) from [MathWorks](https://en.wikipedia.org/wiki/MathWorks" \o "MathWorks), allowing ThingSpeak users to analyze and visualize uploaded data using Matlab without requiring the purchase of a Matlab license from Mathworks.

ThingSpeak has a close relationship with [Mathworks](https://en.wikipedia.org/wiki/Mathworks" \o "Mathworks), Inc. In fact, all of the ThingSpeak documentation is incorporated into the Mathworks' Matlab documentation [site](https://www.mathworks.com/help/thingspeak/) and even enabling registered Mathworks user accounts as valid login credentials on the ThingSpeak website. The terms of service and privacy policy of ThingSpeak.com are between the agreeing user and Mathworks, Inc.

ThingSpeak has been the subject of articles in specialized "[Maker](https://en.wikipedia.org/wiki/Maker_culture" \o "Maker culture)" websites like [Instructables](https://en.wikipedia.org/wiki/Instructables" \o "Instructables), [Codeproject](https://en.wikipedia.org/wiki/The_Code_Project" \o "The Code Project), and [Channel 9](https://en.wikipedia.org/wiki/Channel_9_(Microsoft)" \o "Channel 9 (Microsoft)).

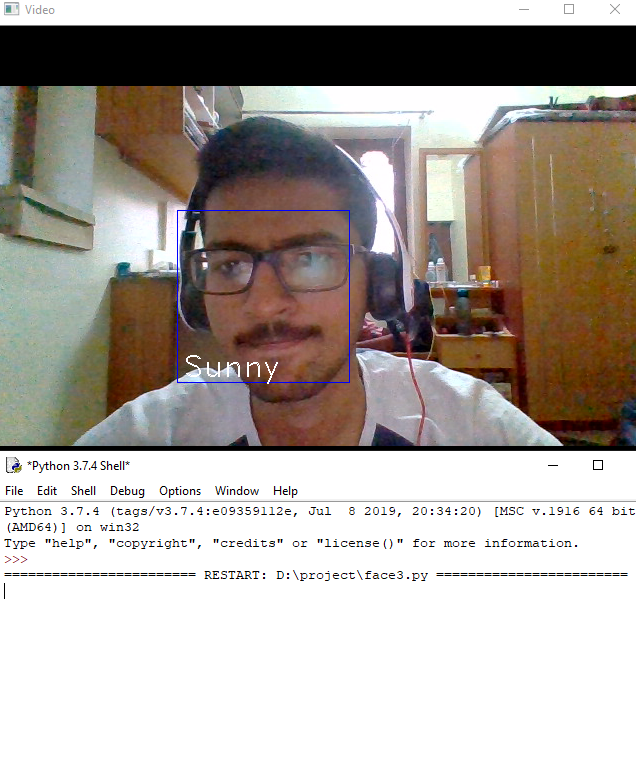
# 3.4.1 ThingSpeak Key Features

ThingSpeak allows you to aggregate, visualize and analyze live data streams in the cloud. Some of the key capabilities of ThingSpeak include the ability to:

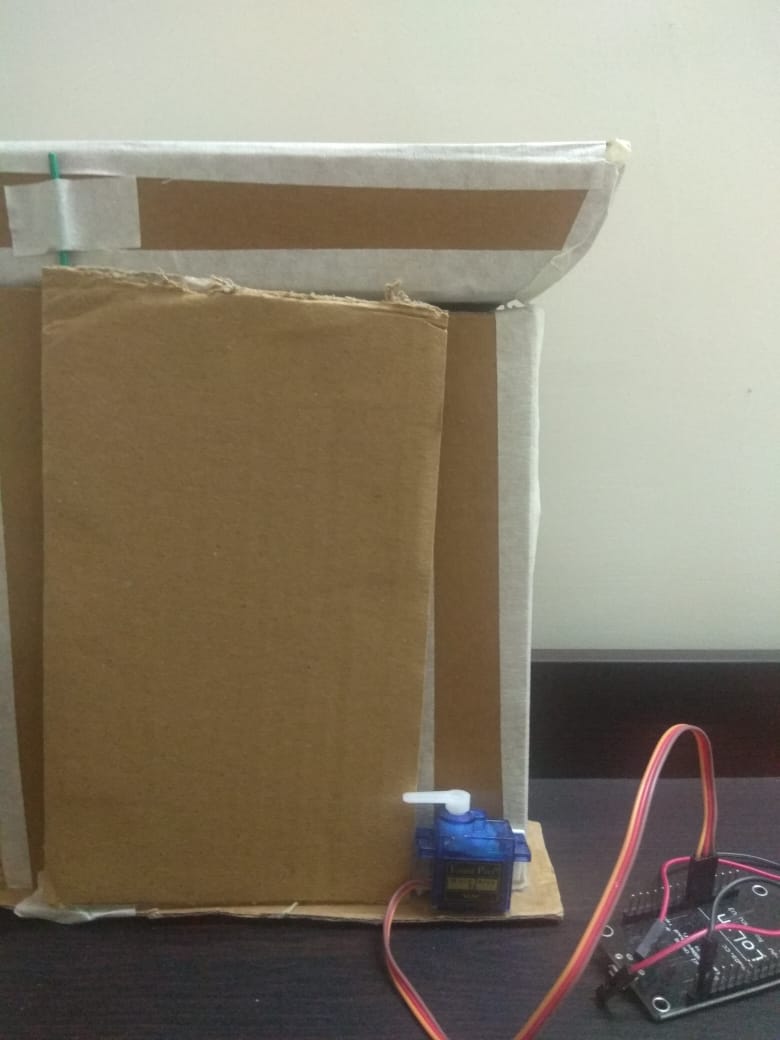
* Easily configure devices to send data to ThingSpeak using popular IoT protocols.
* Visualize your sensor data in real-time.
* Aggregate data on-demand from third-party sources.
* Use the power of MATLAB to make sense of your IoT data.
* Run your IoT analytics automatically based on schedules or events.
* Prototype and build IoT systems without setting up servers or developing web software.
* Automatically act on your data and communicate using third-party services like Twilio or Twitter.

**CHAPTER-4**

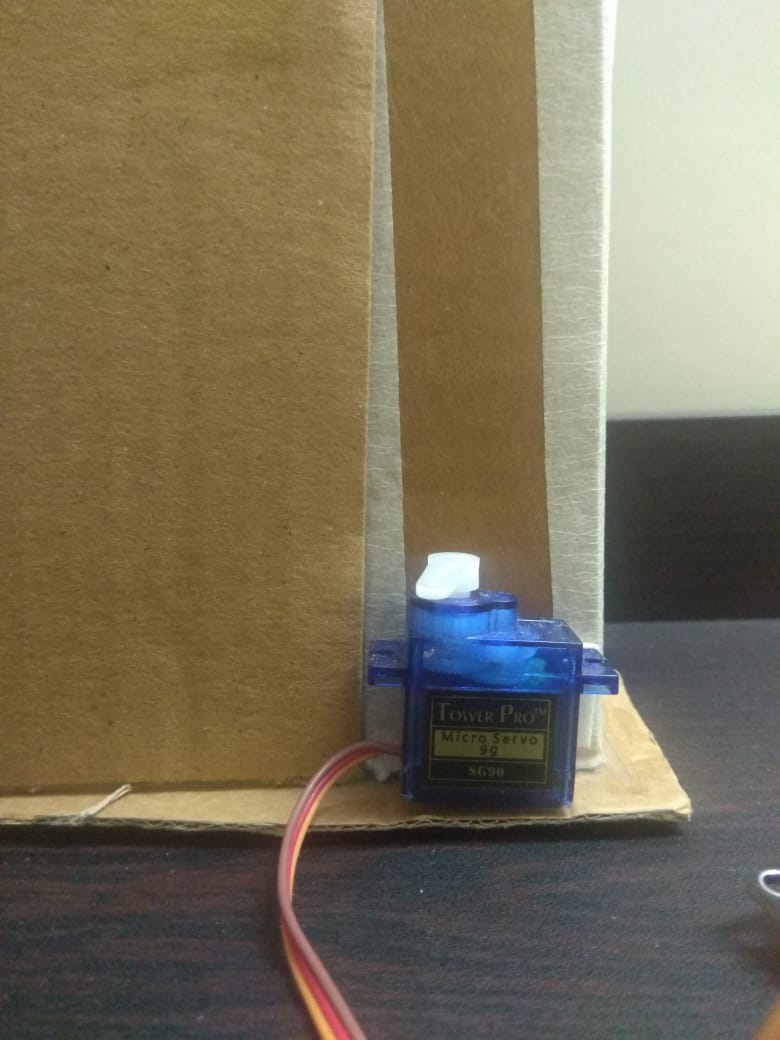
**4.1 RESULTS :**

****

**Fig 4.1.1- face detecting.**



**Fig 4.1.2 - Door closed.**

****

****

**Fig 4.1.3- Door open.**

**CONCLUSION**

In this proposed system door access system by using face recognition has been presented. This system has been tested successfully with home door lock access control based on face recognition method by verifying enrolled facial images.The proposed system is completely standalone and wireless to form a reliable, robust, easily operable, and low price security system. The internet communication has been achieved by connecting through USB cellular data card. The battery power source has been provided to make this whole system as standalone security device successfully. We conclude that various operations are successfully tested and results are documented

**REFERENCES**

1. Hteik Htar Lwin, Aung Soe Khaing, Hla Myo Tun, “Automatic Door Access System UsingFace Recognition”, International Journal of Scientific & Technology Research Volume 4, Issue 06, June 2015.
2. Sadeque Reza Khan, Ahmed Al Mansur, Alvir Kabir, Shahid Jaman, Nahian Chowdhury, “Design and Implementation of Low Cost Home Security System using GSM Network”, International Journal of Scientific & Engineering Research, Volume 3, Issue 3, March 2012.
3. Shambhavi V. Chippa , Dr. R. R. Dube, 2019, AWS EC2 based Home Security System using Face Recognition, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 08, Issue 08 (August 2019).