Chapter 1

INTRODUCTION

The following document is a report on the Group Project on Smart door using facial recognition. It involved building a system for face detection and face recognition using several classifiers available in the open computer vision library(OpenCV). Face recognition is a non-invasive identification system and faster than other systems since multiple faces can be analysed at the same time. The difference between face detection and identification is, face detection is to identify a face from an image and locate the face. Face recognition is making the decision ”whose face is it ? ”, using an image database. In this project both are accomplished using different techniques and are described below. The report begins with a brief history of face recognition. This is followed by the explanation of HAAR-cascades, Eigenface, Fisherface and Local binary pattern histogram (LBPH) algorithms. Next, the methodology and the results of the project are described. A discussion regarding the challenges and the resolutions are described. Finally, a conclusion is provided on the pros and cons of each algorithm and possible implementations.

1.1 History of Facial Recognition

Face recognition began as early as 1977 with the first automated system being introduced By Kanade using a feature vector of human faces. In 1983, Sirovich and Kirby introduced the principal component analysis(PCA) for feature extraction. Using PCA, Turk and Pentland Eigenface was developed in 1991 and is considered a major milestone in technology. Local binary pattern analysis for texture recognition was introduced in 1994 and is improved upon for facial recognition later by incorporating Histograms(LBPH). In 1996 Fisherface was developed using Linear discriminant analysis (LDA) for dimensional reduction and can identify faces in different illumination conditions, which was an issue in Eigenface method. Viola and Jones introduced a face detection technique using HAAR cascades and ADABoost. In 2007, A face recognition technique was developed by Naruniec and Skarbek using Gabor Jets that are similar to mammalian eyes. In This project, HAAR cascades are used for face detection and Eigenface, Fisherface and LBPH are used for face recognition.

1.2 Objective of Study

The robot must be capable of recognizing a human and open the door for him, it also must be connected to a distributed environment with secure toggling of door state.

- It should be capable of recognizing faces and distinguishing them.
- It must be prepared of a situation when there is no face registered but yet the door needs to be open.
- The product must also be capable of recognizing in low light.
- The product must be insensitive to environmental factors such as lighting and noise.
- It must allow calibration of face and darkness threshold.
- The door must be reliable.
- Scalability must be a primary concern in the design.

1.3 Problem Defination

In the present scenario everyone prefers smart objects. Keys of their home or office doors are often lost or are forgotten. Duplicate keys can be subjected to theft and a great threat as they can be easily copied.

1.4 Project Scope

The use of biometrics is one of the solutions but there is a scope for further enhanced products as Facial Recognition enabled door. The world needs comfort and security at figure tips and this product defines the ease of securing the door with biometrics and opening the door for a guest form anywhere in the world.

1.5 Block Diagram

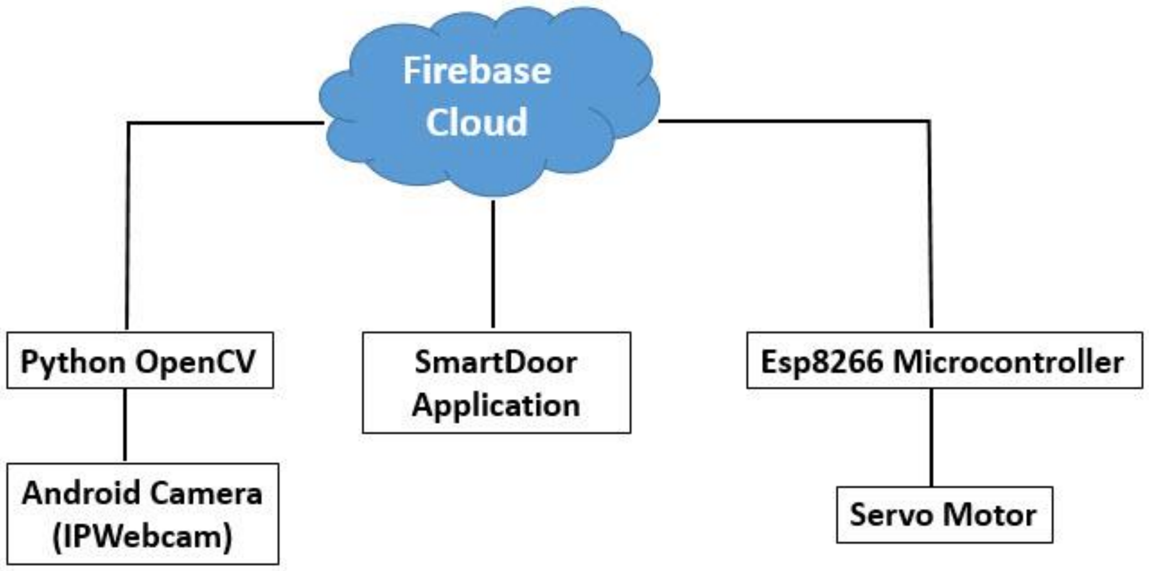


Fig.1.1 Block Diagram

Chapter 2

LITERATURE SURVEY

In recent years a great deal of time and effort has been spent of developing systems to enable an autonomous robot to follow a marked path using a vision system. Not surprisingly, the majority of this research has been towards modifying, or designing from scratch, a full-sized road vehicle so that it can drive on ordinary roads without human supervision. Due to the large amount of space available in an ordinary road vehicle, high performance computers can be used to perform complex image processing and, typic ally, to maintain a mathematical model of the vehicle and the environment.

Forgetting keys is a very common problem in the entire world, to solve there has been many solutions for centuries, yet those solutions are being made smarter. A lock was first invented with a several keys, then came an era of passcode enabled mechanical locks, then those mechanical locks were given electronic abilities and were digitalized. Further RFID locks were invented to which a key was just a card, then digital locks with biometric authentication came into picture. The scope of smart locks is huge in the world.

With a smart lock Security is also concerned, antitheft and surveillance are also great market seeking.

2.1 Autonomous Robots

Autonomous robots are independent of any controller and can act on their own. The robot is programmed to respond in a particular way to an outside stimulus. The bump-and-go robot is a good example. This robot uses bumper sensors to detect obstacle. When the robot is turned on, it moves in a straight direction and when it hits an obstacle, the crash triggers its bumper sensor. The robot gives a programming instruction that asks the robot to back up, turn to the right direction and move forward. This is its response to every bump. In this way, the robot can change direction every time, it encounters an obstacle.

A more elaborate version of the same idea is used by more advanced robots. Robotics create new sensor systems and algorithms to make robots more perceptive and smarter . Today, robots are able to effectively navigate a variety of environments. Obstacle avoidance can be implemented as a reactive control law whereas path planning involves the pre-computation of an obstacle- free path which a controller will then guide a robot along.

Some mobile robots also use various ultrasound sensors to see obstacles or infrared. These sensors work in a similar fashion to animal echolocation. The robot sends out a beam of infrared light or a sound signal. It then detects the reflection of the signal. The robot locates this distance to the obstacles depending on how long it takes the signal to bounce back.

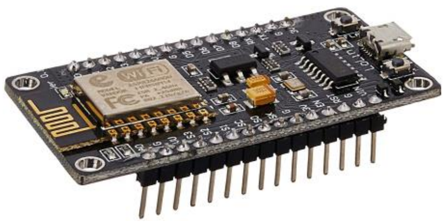
Some advanced robots also use stereo vision. Two cameras provide robots with depth perception. Image recognition software then gives them the ability to locate, classify various objects. Robots also use smell and sound sensors to gain knowledge about its surroundings

2.2 Hardware Components

1. SERVO MOTOR

2.1 Servo Motor SG90

2. MICROCONTROLLER



2.2 NodeMCU ESP8266

3. LEDs

2.3 LED

Chapter 3
STEPS IN SOFTWARE AND CIRCUIT DESIGNING

3.1 Software Tools

3.1.1 Arduino IDE

After the block diagram, coding needs to be done. The software used for coding is avr studio4 and the language used for coding is “embedded c”.

The program code acts as the decision-maker embedded in the microcontroller i.e. it decides what will be the outputs for particular set of input combination. Programs for t he AVR series of microcontrollers can be written in assembly C and AVR Studio etc. We are using winAVR for programming and AVR Studio for simulating (Simulation means debugging the code on software, one can virtually give the input and check the output for that code). In winAVR programmers Notepad we write our C code, after co mpilation it generates ‘.hex’ file that is a hardware level code.

ESP8266 espressif board is used. For this an addition Board is to be installed using .json link and adding the board from board manager.

3.1.2 Python Open CV

OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision.Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license.

We have used python as the scripting language because of its diversity and open source virtue.

3.1.3 Kodular

Kodular (formerly Makeroid) is an online suite for mobile app development. It mainly provides a free drag-and-drop Android app creator without coding, based on

MIT AppInventor. It brings lots of new features like new components and blocks. It also provides a free online app store to share and distribute apps and an extensions IDE for advanced users

What is Kodular?

Kodular allows you to create Android apps easily with a blocks-type editor. No coding skills required. With the Material Design UI, your apps will stand out.

History

Kodular was founded the 6 July 2017 by Conor Shipp, Diego Barreiro, Mika, Pavitra Golchha, Sander Jochems, Sivagiri Visakan and Vishwas Adiga. The first public beta became available on the next month. At that time, Kodular features were nearly the same ones as MIT App Inventor ones, and the interface was not changed a lot.

Kodular was presented at Galicia Maker Faire on September 2017, which was awarded the Makers of Merit prize. In addition, it was invited to InnovAmes where the platform was introduced to the public.

However, on February, Kodular was turned off to the public due to a massive increase on requests. During that time, Kodular became a company in the Netherlands, and it was presented on Galiciencia 2017 where it was widely covered by the press.

The 22nd June 2018 Kodular Creator became online again, which brought lots of new components and features to the platform. Also, the Kodular Store was released, with Kodular Extensions IDE and Kodular Account.

On January 2019 Kodular Creator reached 100,000 active users, and a total of 700,000 unique users per day in apps made with its platform. Half a year later, on August 2019, these number increased up to 200,000 active users in the platform and more than 1,500,000 unique daily users in apps. This makes Kodular Creator one of the largest Android development platforms, having more than 500,000 apps hosted in the platform

Etymology

The word Kodular comes from joining two words: Kode + Modular. The word Kode comes from derivating Code by replacing the C with a K (this was intended as users at Kodular does not use normal Code, they use pseudo-code). Kodular also provides different services, which are called Modules, like the App Store and the Extensions IDE.

Also, it is being said that Kodular in Estonian means "the core", which the purpose of being the core of introduction to coding.

Kodular Creator (formerly known as Makeroid) is the main web app and the core of all services. It is built primarily for newcomers to computer programming to create software applications for the Android operating system (OS). It has an user graphical interface, similar to Scratch, MIT App Inventor, and its distributions. Only by dragging and dropping a few

components, and joining some blocks as in Scratch the app is made. It runs on Google App Engine, using several Google Cloud services.

3.1.4 Firebase Cloud

Firebase provides a real-time database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. The company provides client libraries that enable integration
with Android, iOS, JavaScript, Java, Objective-C, Swift and Node.js applications. The database is also accessible through a REST API and bindings for
several JavaScript frameworks such
as AngularJS, React, Ember.js and Backbone.js. The REST API uses the Server-Sent Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the real-time database can secure their data by using the company's server-side-enforced security rules.

3.2 Face Recognition Process

For this project three algorithms are implemented independently. These are Eigenface, Fisherface and Linear binary pattern histograms respectively. All three can be implemented using OpenCV libraries. There are three stages for the face recognition as follows:

1. Collecting images IDs

2. Extracting unique features, classifying them and storing in XML files

3. Matching features of an input image to the features in the saved XML files and predict identity

3.2.1 Collecting the image data:

Collecting classification images is usually done manually using a photo editing software to crop and resize photos. Furthermore, PCA and LDA requires the same number of pixels in all the images for the correct operation. This time consuming and a laborious task is automated through an application to collect 50 images with different expressions. The application detects suitable expressions between 300ms, straightens any existing tilt and save them.

3.2.2 Training the Classifiers:

OpenCV enables the creation of XML files to store features extracted from datasets using the FaceRecognizer class. The stored images are imported, converted to grayscale and saved with IDs in two lists with same indexes. FaceRecognizer objects are created using face recogniser class.

cv2.face.createLBPHFaceRecognizer():

1. The radius from the centre pixel to build the local binary pattern.

2. The Number of sample points to build the pattern. Having a considerable number will slow down the computer.

3. The Number of Cells to be created in X axis.

4. The number of cells to be created in Y axis.

5. A threshold value similar to Eigenface and Fisherface. if the threshold is passed the object will return -1

3.2.3 The Face Recognition:

Face recogniser object is created using the desired parameters. Face detector is used to detect faces in the image, cropped and transferred to be recognised. This is done using the same technique used for the image capture application. For each face detected, a prediction is made using FaceRecognizer.predict() which return the ID of the class and confidence. The process is same for all algorithms and if the confidence his higher than the set threshold, ID is -1. Finally, names from the text file with IDs are used to display the name and confidence on the screen. If the ID is -1, the application will print unknown face without the confidence level.

Source Code:

Arduino IDE:

```
#include <ESP8266WiFi.h>
#include<FirebaseArduino.h>
#include <Servo.h>

#define FIREBASE_HOST
"smartdoorgp.firebaseio.com"
#define FIREBASE_AUTH
"xWdXdam02D7Tkxgkbi55iiHQta73ecc0HKaqn
D67"

const char* ssid = "MM";
const char* password = "nopassword";

//const int trigPin = D6;
//const int echoPin = D7;
int state=0, pos;
long duration;
int distance;
String Door,Face;
Servo myservo;


void setup() {
  Serial.begin(9600);
  //off();
  Serial.println("Connecting");
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.waitForConnectResult() !=
WL_CONNECTED) {
    Serial.println("Connection Failed!
Rebooting...");
    delay(5000);
    ESP.restart();
  }
  Serial.println("Connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  //pinMode(trigPin, OUTPUT);
  //pinMode(echoPin, INPUT);
  myservo.attach(D1);
  Firebase.begin(FIREBASE_HOST,
FIREBASE_AUTH);
 }

 void loop() {
  //verification();
  Door =
Firebase.getString("SmartDoor/Door");
  Face =
Firebase.getString("SmartDoor/Face");

  if(Door=="\"Open\"" && state==1){
    Open();
    Door="";
    Firebase.setString("SmartDoor/Door",
"");
  }
  if(Door=="\"Close\"" && state==0){
    Close();
    Door="";
    Firebase.setString("SmartDoor/Door",
"");
  }
  if(Face=="mm" && state==1){
    Open();
    delay(5000);
    Close();
    Face="";
    Firebase.setString("SmartDoor/Face",
"");
  }
  }

/*  void off(){
  digitalWrite(trigPin,LOW);
  digitalWrite(echoPin,LOW);
}



void verification(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance= duration*0.034/2;
  Serial.print("Wall Distance: ");
  Serial.println(distance);
```

```
   if(distance<=2){
     //Already Open
     state=0;
     Firebase.setString("SmartDoor/State",
"Door is Open");

   }
   else{
     //Closed
     state=1;
     Firebase.setString("SmartDoor/State",
"Door is Close");
   }
 }*/

 void Open(){
   for (pos = 50; pos <= 180; pos += 1) { //
goes from 0 degrees to 180 degrees
     // in steps of 1 degree
     myservo.write(pos);              // tell servo
to go to position in variable 'pos'
     delay(15);                       // waits 15ms
for the servo to reach the position
     Serial.println(pos);
   }
   state=0;
   Firebase.setString("SmartDoor/State",
"Door is Open");
 }
 void Close(){
   for (pos = 180; pos >= 50; pos -= 1) { //
goes from 180 degrees to 0 degrees
     myservo.write(pos);              // tell servo
```

```
      to go to position in variable 'pos'
          delay(15);                       // waits 15ms
      for the servo to reach the position
          Serial.println(pos);
        }
        state=1;
        Firebase.setString("SmartDoor/State",
      "Door is Close");
       }
```

OpenCV:

dataSetGenerator.py:

```
import cv2

import os

import urllib

import imutils


path =
os.path.dirname(os.path.abspath(__file__))

detector=cv2.CascadeClassifier(path+r'\Class
ifiers\face.xml')

i=0

offset=50

name=raw_input('enter your id')


#cam = cv2.VideoCapture(0)

while True:

   #Through IP webcam

   IP1 = 'http://192.168.43.1:8080/shot.jpg'

   urllib.urlretrieve(IP1, 'shot1.jpg')

   frame = cv2.imread('shot1.jpg')

   frame = imutils.resize(frame, width=600)
```

```
    #Through Laptop Cam
    #ret, frame =cam.read()

gray=cv2.cvtColor(frame,cv2.COLOR_BGR
2GRAY)
    faces=detector.detectMultiScale(gray,
scaleFactor=1.2, minNeighbors=5,
minSize=(100, 100),
flags=cv2.CASCADE_SCALE_IMAGE)
    for(x,y,w,h) in faces:
        i=i+1
        cv2.imwrite("dataSet/face-"+name +'.'+
str(i) + ".jpg", gray[y-offset:y+h+offset,x-
offset:x+w+offset])
        cv2.rectangle(frame,(x-50,y-
50),(x+w+50,y+h+50),(225,0,0),2)
        cv2.imshow('frame',frame[y-
offset:y+h+offset,x-offset:x+w+offset])
        cv2.waitKey(100)
    if i>20:
        #cam.release()
        cv2.destroyAllWindows()
        break
```

trainer.py:

```
import cv2,os
import numpy as np
from PIL import Image

cascade =
cv2.CascadeClassifier('cascades/data/haarcas
cade_frontalface_alt2.xml')
path =
os.path.dirname(os.path.abspath(__file__))
```

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

cascadePath = path+r"\Classifiers\face.xml"

faceCascade =
cv2.CascadeClassifier(cascadePath);

dataPath = path+r'\dataSet'


def get_images_and_labels(datapath):

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

    # images will contains face images

    images = []

    # labels will contains the label that is
assigned to the image

    labels = []

    for image_path in image_paths:

        # Read the image and convert to
grayscale

        image_pil =
Image.open(image_path).convert('L')

        # Convert the image format into numpy
array

        image = np.array(image_pil, 'uint8')

        # Get the label of the image

        nbr =
int(os.path.split(image_path)[1].split(".")[0].r
eplace("face-", ""))

        #nbr=int(''.join(str(ord(c)) for c in nbr))

        print(nbr)

        # Detect the face in the image

        faces =
faceCascade.detectMultiScale(image)

        # If face is detected, append the face to
images and the label to labels
```

```
        for (x, y, w, h) in faces:
            images.append(image[y: y + h, x: x +
w])
            labels.append(nbr)
            cv2.imshow("Adding faces to traning
set...", image[y: y + h, x: x + w])
            cv2.waitKey(10)
    # return the images list and labels list
    return images, labels



images, labels =
get_images_and_labels(dataPath)
cv2.imshow('test',images[0])
cv2.waitKey(1)

recognizer.train(images, np.array(labels))
recognizer.save(path+r'\trainer\trainer.yml')
cv2.destroyAllWindows()
```

detector.py:

```
import cv2,os
import numpy as np
from PIL import Image
from firebase import firebase
import urllib
import imutils

firebase =
firebase.FirebaseApplication('https://smartdo
orgp.firebaseio.com/')
```

```
path =
os.path.dirname(os.path.abspath(__file__))

face_cascade =
cv2.CascadeClassifier('cascades/data/haarcas
cade_frontalface_alt2.xml')


recognizer =
cv2.face.LBPHFaceRecognizer_create()

recognizer.read(path+r'\trainer\trainer.yml')

cascadePath = path+"\Classifiers\face.xml"

faceCascade =
cv2.CascadeClassifier(cascadePath);


#cam = cv2.VideoCapture(0)

font = cv2.FONT_HERSHEY_SIMPLEX
#Creates a font

while True:

    #Through IP webcam

    IP1 = 'http://192.168.43.1:8080/shot.jpg'

    urllib.urlretrieve(IP1, 'shot1.jpg')

    frame = cv2.imread('shot1.jpg')

    frame = imutils.resize(frame, width=600)


    #Through Laptop Cam

    #ret, frame =cam.read()


gray=cv2.cvtColor(frame,cv2.COLOR_BGR
2GRAY)

    faces =
face_cascade.detectMultiScale(gray,
scaleFactor=1.5, minNeighbors=5)

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

        nbr_predicted, conf =
recognizer.predict(gray[y:y+h,x:x+w])
```

```
        cv2.rectangle(frame,(x-50,y-
50),(x+w+50,y+h+50),(0,255,0),2)
        if(nbr_predicted==0 and conf<=63):
            tag='mm'
        if(nbr_predicted==1 and conf<=63):
            tag='venky'
        if(nbr_predicted==2 and conf<=63):
            tag='sanju'
        else:
           tag='unknown'
        cv2.putText(frame,tag, (x,y),font, 1.1,
(255,0,0), 2, cv2.LINE_AA) #Draw the text
        firebase.put('/SmartDoor','Face',tag)
        print(tag)
        print(conf)


    cv2.imshow('frame',frame)
    k=cv2.waitKey(20) & 0xFF
    if k==27 or k==ord('q'):
        break


# When everything done, release the capture
#cam.release()
cv2.destroy
```

Chapter 4

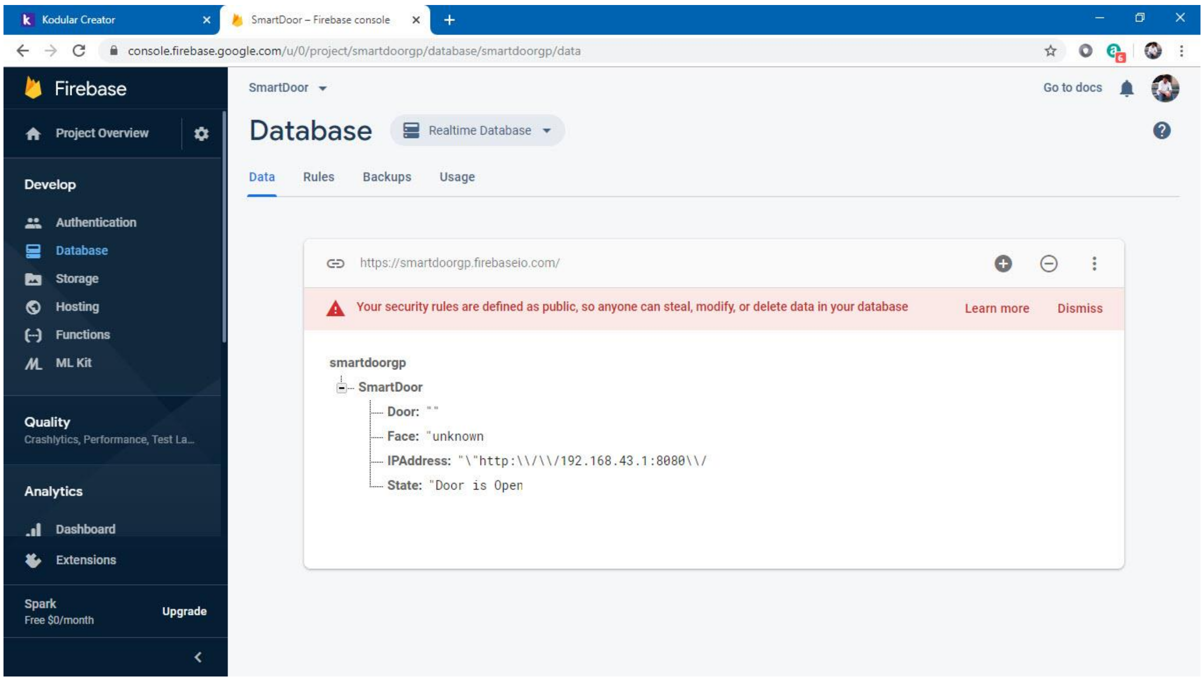
RESULT & DISCUSSION

Our project is an innovative idea of intelligent system which has basically face detection feature and will provide help in various places.

1. The battery activates the circuit.
2. The laptop is switched on and the program runs, it sends the information to the cloud. Or The application in the android can toggle the door state.
3. Esp8266 receives the data using internet connected hotspot.
4. Servo motor rotates in accordance to the signal given by the microcontroller

4.1 Result

The objective of the Smart door is to provide security at ease. We have been successful in providing the same. The door is successfully designed and built, the product is efficient and works best with a good internet connectivity.



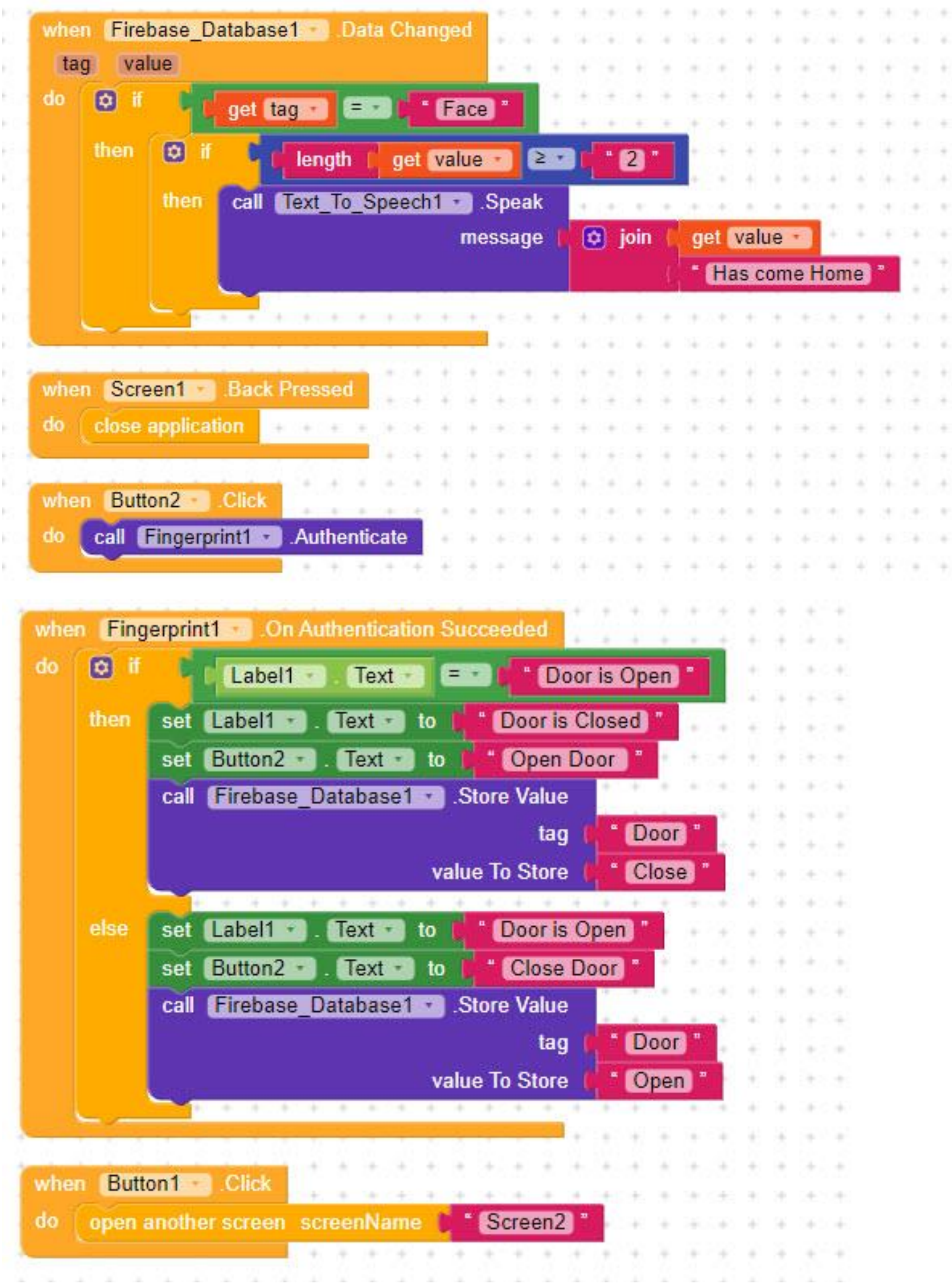
console.firebase.google.com/u/0/project/smartdoorgp/database/smartdoorgp/data

https://smartdoorgp.firebaseio.com/

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

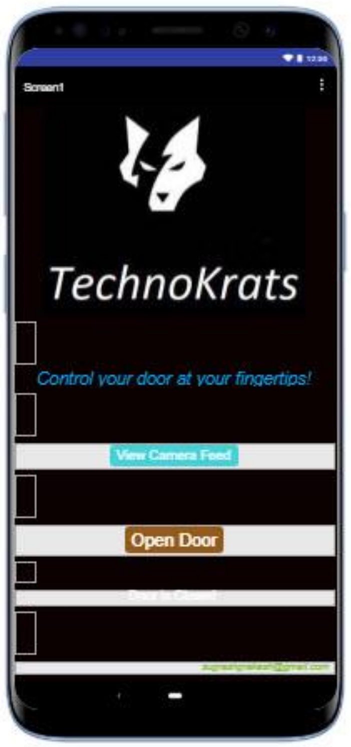
```
smartdoorgp
  SmartDoor
    Door: ""
    Face: "unknown
    IPAddress: "\"http:\\/\\/192.168.43.1:8080\\/
    State: "Door is Open
```

4.1 Output Firebase Database



```
when Firebase_Database1 .Data Changed
  tag  value
do  if  get tag = "Face"
    then  if  length get value ≥ "2"
          then  call Text_To_Speech1 .Speak
                  message  join  get value
                                 "Has come Home"

when Screen1 .Back Pressed
do  close application

when Button2 .Click
do  call Fingerprint1 .Authenticate

when Fingerprint1 .On Authentication Succeeded
do  if  Label1 . Text = "Door is Open"
    then  set Label1 . Text to "Door is Closed"
          set Button2 . Text to "Open Door"
          call Firebase_Database1 .Store Value
                       tag  "Door"
              value To Store  "Close"
    else  set Label1 . Text to "Door is Open"
          set Button2 . Text to "Close Door"
          call Firebase_Database1 .Store Value
                       tag  "Door"
              value To Store  "Open"

when Button1 .Click
do  open another screen  screenName  "Screen2"
```

4.2 Output Kodular Blocks



4.3 Output Application

4.2 Advantages & Disadvantages

Advantages

- Door movement is automatic.
- Fit and Forget system.
- Cost effective.
- Simplicity of building

Disadvantages

- Needs Internet always
- Sophisticated processors for Image processing may lead to cost increment.
- Electricity requirement is a must.

4.3 Applications

- Class Room doors with automated attendance system.
- Industrial applications.
- Home applications.

Chapter 5
CONCLUSION AND FUTURE SCOPE OF WORK

5.1 Conclusion

In this project we have studied and implemented a Smart Door using Facial Recognition and an application with a virtue of IoT for general people. The programming and interfacing of microcontroller has been mastered during the implementation.

Nowadays every gadget being used is turning into a smart gadget, everything we use is turning into an electronic gadget. A little bit of programming into a Microcontroller or Microprocessor will make an electronic gadget a smarter tool. The use of technology is now not just restricted to Space agencies or laboratories but also a home is now equally technically equipped right form the entrance. Our Idea of SMART DOOR is successful in its idea of making life at home more comfortable and giving our home an ability to recognize us.

5.2 Future Scope

Use of enhanced processors for processing instead of the Laptop being on forever. The present system processes the Image data and sends it to a cloud and then it is sent to the Microcontroller on the door to open it. This is a time consuming and tedious task, instead we can inculcate the entire task inside the door using the processors.

Chapter 6

BIBLOGRAPHY

6.1 References

[1] Takeo Kanade. Computer recognition of human faces, volume 47. Birkh¨auser Basel, 1977.

[2] Lawrence Sirovich and Michael Kirby. Low-dimensional procedure for the characterization of human faces. Josa a, 4(3):519–524, 1987.

[3] https://www.instructables.com/id/NodeMCU-ESP8266-Details-and-Pinout/

[4] Dong chen He and Li Wang. Texture unit, texture spectrum, and texture analysis. IEEE Transactions on Geoscience and Remote Sensing, 28(4):509–512, Jul 1990.

[5] X. Wang, T. X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In 2009 IEEE 12th International Conference on Computer Vision, pages 32–39, Sept 2009.

[6] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7):711–720, Jul 1997.

[7] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, volume 1, pages I–511–I–518 vol.1, 2001.

[8] John G Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. JOSA A, 2(7):1160–1169, 1985.

[9] S Marˆcelja. Mathematical description of the responses of simple cortical cells. JOSA, 70(11):1297– 1300, 1980.

[10] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In Image Processing. 2002. Proceedings. 2002 International Conference on, volume 1, pages I–I. IEEE, 2002.

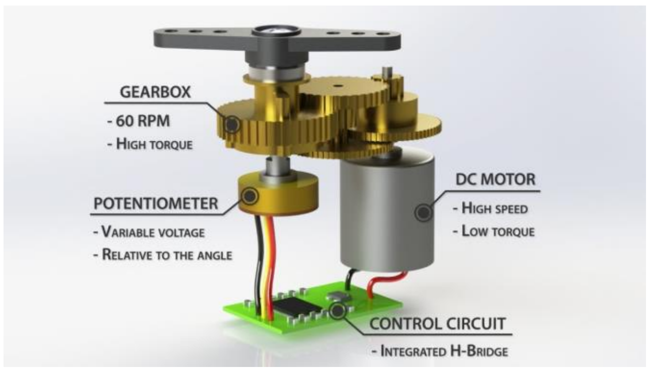
[11] John P Lewis. Fast template matching. In Vision interface, volume 95, pages 15–19, 1995.

Chapter 7

APPENDIX

7.1 Servo Motor (SG90)

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.



7.1 Servo Motor Internal

Servo Motor Construction:

The following figure shows the construction of a standard servo motor. ISL Products offers a wide variety of servo motors to accommodate for different industrial applications. The servo motor consists of two winding stator and rotor windings. The stator winding is wound on the stationary part of the motor, and this winding is also called field winding of the motor. The rotor winding is wound on the rotating part of the motor and this winding is also called the armature winding of the motor. The motor consists of two bearings on front and back side for the free movement of shaft. The encoder has the approximate sensor for determining rotational speed and revolution per minute of the motor. Our Motor Selection Tool will help you identify your requirements.

Servo Mechanism

It consists of three parts:

1. Controlled device
2. Output sensor
3. Feedback system

Working of servo motors :

Servo motors control position and speed very precisely. Now a potentiometer can sense the mechanical position of the shaft. Hence it couples with the motor shaft through gears. The current position of the shaft is converted into electrical signal by potentiometer, and is compared with the command input signal. In modern servo motors, electronic encoders or sensors sense the position of the shaft.

We give command input according to the position of shaft. If the feedback signal differs from the given input, an error signal alerts the user. We amplify this error signal and apply as the input to the motor, hence the motor rotates. And when the shaft reaches to the require position, error signal become zero, and hence the motor stays standstill holding the position.

The command input is in form of electrical pulses. As the actual input to the motor is the difference between feedback signal (current position) and required signal, hence speed of the motor is proportional to the difference between the current position and required position. The amount of power require by the motor is proportional to the distance it needs to travel.

Principle of working:

Servo motor works on the PWM (Pulse Width Modulation) principle, which means its angle of rotation is controlled by the duration of pulse applied to its control PIN. Basically servo motor is made up of DC motor which is controlled by a variable resistor (potentiometer) and some gears

Mechanism of servomotor:

Basically a servo motor is a closed-loop servomechanism that uses position feedback to control its motion and final position. Moreover the input to its control is a signal (either analogue or digital) representing the position commanded for the output shaft .

The motor is incorporates some type of encoder to provide position and speed feedback. In the simplest case, we measure only the position. Then the measured position of the output is compared with the command position, the external input to controller. Now If the output position differs from that of the expected output, an error signal generates.

Which then causes the motor to rotate in either direction, as per need to bring the output shaft to the appropriate position. As the position approaches, the error signal reduces to zero. Finally the motor stops.

The very simple servomotors can position only sensing via a potentiometer and bang-bang control of their motor. Further the motor always rotates at full speed. Though this type of servomotor doesn’t have many uses in industrial motion control, however it forms the basis of simple and cheap servo used for radio control models.

Servomotors also find uses in optical rotary encoders to measure the speed of output shaft and a variable-speed drive to control the motor speed. Now this, when combined with a PID control algorithm further allows the servomotor to be in its command position more quickly and more precisely with less overshooting .

Controlling of servomotors:

Usually a servomotor turns 90 degree in either direction hence maximum movement can be 180 degree. However a normal servo motor cannot rotate any further to a build in mechanical stop.

We take three wires are out of a servo: positive , ground and control wire. A servo motor is control by sending a pulse width modulated (PWM) signal through the control wire. A pulse is sent every 20 milliseconds. Width of the pulses determine the position of the shaft.

Applications:

1. Robotics: At every joint of the robot, we connect a servomotor. Thus giving the robot arm its precise angle.

2. Conveyor belts: servo motors move, stop, and start conveyor belts carrying product along to various stages, for example, in product packaging/ bottling, and labelling.

3. Camera auto focus: A highly precise servo motor build into the camera corrects a camera lens to sharpen out of focus images.

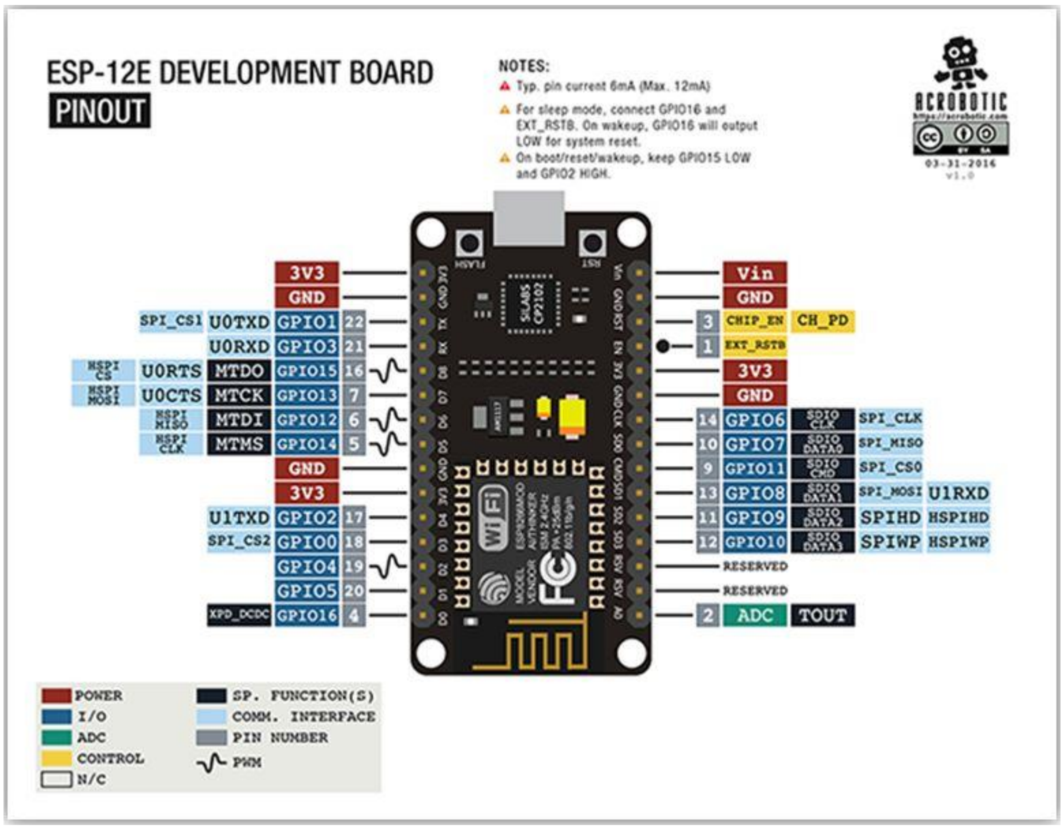
4. Solar tracking system: Servo motors adjust the angle of solar panels throughout the day and hence each panel continues to face the sun which results in harnessing maximum energy from sunup to sundown.

Advantages:

1. If a heavy load is placed on the motor, the driver will increase the current to the motor coil as it attempts to rotate the motor. Basically, there is no out-of-step condition.

2. High-speed operation is possible.

7.2 NodeMCU (ESP2866):



ESP-12E DEVELOPMENT BOARD
PINOUT
NOTES:
Typ. pin current 6mA (Max. 12mA)
For sleep mode, connect GPIO16 and EXT_RSTB. On wakeup, GPIO16 will output LOW for system reset.
On boot/reset/wakeup, keep GPIO15 LOW and GPIO2 HIGH.
ACROBOTIC
3V3
GND
SPI_CS1 U0TXD GPIO1 22
U0RXD GPIO3 21
HSPI CS U0RTS MTDO GPIO15 16
HSPI MOSI U0CTS MTCK GPIO13 7
HSPI MISO MTDI GPIO12 6
HSPI CLK MTMS GPIO14 5
GND
3V3
U1TXD GPIO2 17
SPI_CS2 GPIO0 18
GPIO4 19
GPIO5 20
XPD_DCDC GPIO16 4
Vin
GND
3 CHIP_EN CH_PD
1 EXT_RSTB
3V3
GND
14 GPIO6 SDIO CLK SPI_CLK
10 GPIO7 SDIO DATA0 SPI_MISO
9 GPIO11 SDIO CMD SPI_CS0
13 GPIO8 SDIO DATA1 SPI_MOSI U1RXD
11 GPIO9 SDIO DATA2 SPIHD HSPIHD
12 GPIO10 SDIO DATA3 SPIWP HSPIWP
RESERVED
RESERVED
2 ADC TOUT
POWER
I/O
ADC
CONTROL
N/C
SP. FUNCTION(S)
COMM. INTERFACE
PIN NUMBER
PWM

7.2 NodeMCU Pinout