INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 7, Issue 4 - V7I4-1269)
Available online at: https://www.ijariit.com

# Development of a home security Robot using Deep Learning and IoT

*Shravan Aruljothi*
*shrav.aruljothi@gmail.com*
*New Horizon College of Engineering, Bengaluru, Karnataka*

*Sharad Dewanand Parate*
*sharad1042@gmail.com*
*New Horizon College of Engineering, Bengaluru, Karnataka*

*Harshit S.*
*harshitshivakumar101999@gmail.com*
*New Horizon College of Engineering, Bengaluru, Karnataka*

*Nehal Dinesh Andani*
*nehal.andani@gmail.com*
*New Horizon College of Engineering, Bengaluru, Karnataka*

## ABSTRACT

***The major problem in every urban city is the lack of security to residential areas. The number of thefts, electricity and food wastage at homes in urban areas increase every year due to human error. As per the National Crime Records Bureau (NCRB), 2,44,119 cases of robbery, theft, and burglary took place in residential premises in 2019. Also, electricity consumption in Indian homes has tripled since 2010. In 2019, an urban Indian household consumed about 90 units (kWh) of electricity as a monthly average which is one-third of the monthly world average. To solve these issues, we have proposed an idea of a "Home Security Robot" for a smart city using AI. The Home Security Robot will help in eliminating the reliance on security guards and will effectively monitor everything in the house (if there are any gas leakage, fridge malfunctions, unnecessary electricity wastage, indoor air quality and any unknown movements inside the house). If the owner is under attack, he/she can shout out "HELP" or "SAVE ME "so that the robot can take in the voice command to automatically call the police. The navigation part is done by Arduino and Bluetooth RC Controller App. There are 2 parts (Face Detection & Recognition using Raspberry Pi and IoT system using BOLT module with sensors).The first part has three python programs used for facial detection and recognition using OpenCV with Haar Cascade Classifier and LBPH algorithm. The first program (Face Dataset) is used for collecting images of known users and storing it in a database using Haar Cascade Classifier. The second program (Face Training) is used to train the stored images using LBPH algorithm so the model can distinguish between the users whose faces are stored in database and then these trained images are stored in the trainer.yml file. The third program (Face Recognition) is used to read the trained images stored in the trainer.yml file and then uses Haar Cascade Classifier to recognize the detected face and identifies whether the face belongs to a user or an intruder. The IoT system with the help of BOLT module helps in checking the temperature in the room and checking if any unnecessary lights are on in the room. If room temperature is outside the safe range specified or if any lights are on, owner will get an alert via SMS.***

***Keywords***— *IoT, Deep Learning, Raspberry Pi, BOLT, OpenCV, Haar Cascade Classifier, Viola Jones algorithm, LBPH algorithm*

## 1. OBJECTIVES

- To apply deep learning with python programming on Raspberry Pi using Open CV algorithm for image classification and facial detection and recognition.
- To develop an IOT system consisting of Raspberry Pi, BOLT Iot Module, sensors (LM35 temperature sensor, LDR, Pi Camera, MQ135 gas sensor) connected with BOLT cloud for effective security and safety of homes and families.

## 2. LITERATURE REVIEW

### 2.1. Comparison of OpenCV's Feature Detectors and Feature matcher (Frazer K. Noble):

This paper describes the implementation and comparison of a range of the library's feature detectors and feature matchers. The study shows that the Speeded-Up Robust Features (SURF) detector finds the greatest number of features in an image, and that the Brute Force (BF) matcher matches the greatest number of detected features in an image pair. On average, they detected and matched 1132.00 features each in 265. 67ms. In a benchmark image set, OpenCV's SURF detector discovered, on average, 1907.20 features in 1538.61 MS, and OpenCV's BF matcher, on average, matched features in 160. 24MS.OpenCV's binary robust invariant scalable

key-points (BRISK) detection and BF matching algorithms were found to be the most efficient feature detectors and matching algorithms on average. On average, they detected and matched 1132.00 features each in 265. 67ms. Based on our analysis, if detecting the number of features is important, the SURF detector should be used, otherwise the BF matcher should be used.

**2.2. Vehicle Theft Tracking, Detecting And Locking Using Open CV System (Mohanasundaram S):**
Based on the vehicle theft tracking and detection system, they have added facial recognition technology. The software provides an ultimate solution to all of these problems. Face recognition technology is used here. The locking and/or detection system(s) installed on a vehicle. Using a mobile application, it recognizes a user's face and compares it with their data to determine if the user belongs to an automated system (or not). The vehicle is unlocked if the condition is true. If any vehicle is locked, the message and call are automatically sent to the responsible person. By using this system, users can prevent vehicles from being stolen as well as view theft details, highlighting theft details and storing them in a USB drive. Position, pose, lighting, background, camera quality, and gender are some of the parameters.

**2.3. Face Recognition system based on LBPH Algorithm (Abhishek Pratap Singh, SunilKumar S Manvi, Pratik Nimbal, Gopal Krishna Shyam, IJEAT)**:
In this paper, we present the Haar cascade Classifier and LBPH algorithm for facial detection and recognition. For the training images, our system uses PyCharm IDE in combination with a SQL database. The results show that the system can recognize both known and unknown individuals. As described in the proposed method, if the rate of change in the frame is too high and there is not enough light while taking the picture, the system will not recognize faces reliably.

**2.4. LBPH Algorithm for Frontal and Side profile Face Recognition on GPU (Ms.Akshata Uday Naik, Dr. Nitheh Guinde:**
We present a GPU implementation of frontal and side profile face recognition using the LBPH algorithm. The performance of the CPU and GPU is then compared. As CPUs require more memory than GPUs, CPU time was found to be greater than GPU time. CPUs are more suitable for serial instructions, while GPUs can process parallel instructions, so GPUs process data faster.

## 3. EXPERIMENTAL EQUIPMENT AND INSTRUMENTATION

### 3.1 System Specification

**3.1.1 Raspberry Pi:** Raspberry Pi is a credit-card sized laptop able to controlling robots and different digital devices. We have used it to program the robot for facial detection and recognition.

**Advantages of raspberry pi:**

- We have used raspberry PI instead of Arduino has it helps us in multitasking
- Has more number of input ports
- 40 GPIO pins which helps us connect more number of components
- It is faster and more reliable
- It has its own built in IDE – THONNY PYTHON
- It can be remotely accessed through any part of the world using SSH and VNC viewer

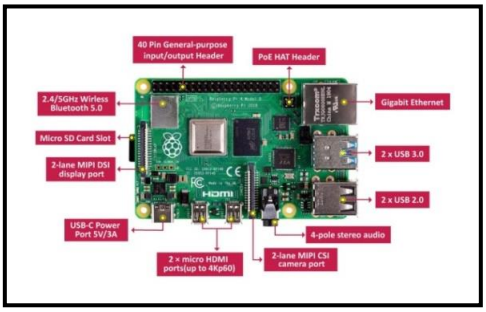



**Fig. Raspberry Pi**

**3.1.2. BOLT IoT:** BOLT is an Internet of Things platform (Hardware + Cloud) that permits consumer to construct IoT merchandise and projects. Using BOLT, customers can manipulate and screen gadgets from any a part of the world.
ADVANTAGES:
• It has its very own cloud known as BOLT Cloud for records visualization and records storage
• No programming required to attach sensors to BOLT.
• It can used for packages like SMS alert the use of TWILIO or TELEGRAM.
• It has its very own cellular app to setup the BOLT and to look results.


**Fig. BOLT Iot**

**3.1.3. LM35 sensor:**

The LM35 collection rectangular degree exactness integrated-circuit temperature gadgets with companion output voltage linearly-proportional to the Centigrade temperature. ...

The LM35 tool is rated to paintings over a −55°C to 150°C temperature vary, while the LM35C tool is rated for a −40°C to 110°C vary (−10° with stepped forward accuracy).

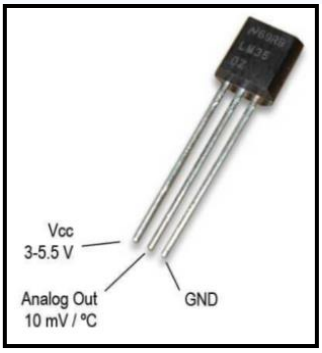



**Fig. LM35 sensor**

**3.1.4. Raspberry PI camera module:**

Raspberry Pi at gift promote styles of digital digicam board: a 8MP system and a 12MP High Quality (HQ) digital digicam. The 8MP system is moreover handy in NoIR shape with out an IR channel. The first 5MP system is not, at this factor handy from Raspberry Pi. The details of the multitude of devices may be located here. All Raspberry Pi cameras are ready for taking high-aim photos, along complete HD 1080p video, and may be absolutely managed automatically. This documentation depicts how to make use of the digital digicam in special situations, and the way to make use of the special programming apparatuses.

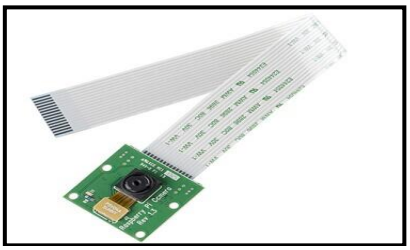


**Fig. Raspberry Pi camera module**

**3.1.5. Arduino Uno**

The Arduino Uno is primarily based totally at the Microchip ATmega328P microcontroller and evolved through Arduino.cc.

The board is prepared with units of virtual and analog input/output (I/O) pins that can be interfaced to numerous enlargement boards (shields) and different circuits. The board has 14 virtual I/O pins (six able to PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), through a kind B USB cable. Arduino is used for navigating the robot with the help of Bluetooth module. The robot communicates through an app where there are several options on the UI to move the robot.

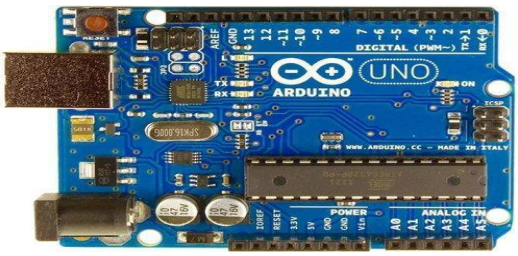



**Fig. Arduino**

**3.2. Robot Parts description:**

**3.2.1 Robot chassis:**

The robotic chassis bares of all of the additives required to run the robotic and additionally affords structural help to the robotic. We designed the chassis the use of CATIA software program and examined the weight bearing ability the use of ANSYS software program.

**3.2.2. L298D Motor Driver**:

L293D is a regular Motor motive force or Motor Driver IC which lets in DC motor to pressure on both directions. L293D is a 16-pin IC that can manipulate a fixed of DC cars concurrently in any direction. Here, you can manipulate DC motor with an unmarried L293D IC.

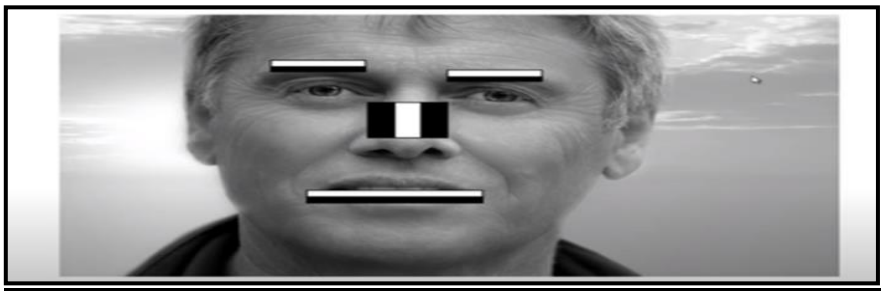
**3.2.3. LIPO Battery**:

A lithium polymer battery, or extra efficaciously lithium-ion polymer Battery is a chargeable battery of lithium-ion using a polymer electrolyte in preference to a liquid electrolyte. High conductivity semisolid (gel) polymers shape this electrolyte. These batteries offer better precise electricity than different lithium battery kinds and are utilized in packages in which weight is a vital characteristic, together with cellular devices, radio-managed plane and a few electric powered vehicles.

## 4. METHODOLOGY

### 4.1 Haarcascade Classifier



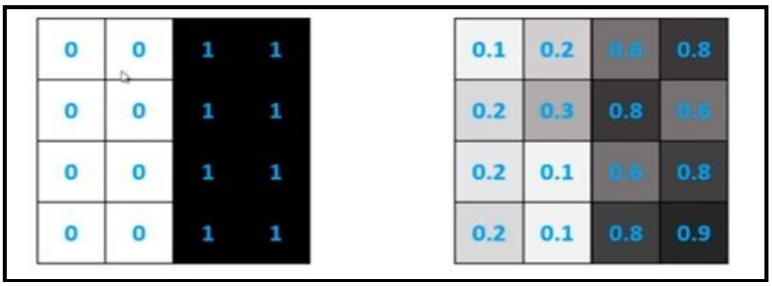
The above image gives a good example of how line feature and edge feature are used. Here the eyebrows is using a edge feature and the nose is using a line feature.

**Working of Haar-cascade classifier**

Object detection using haar feature based cascade classifiers is an effective objection detection method proposed by Paul Viola and Michel Jones. It is a deep learning based approach where a cascade function is gained from positive and negative images. It is then used to detect objects in other images.Initially,rhe algorithm needs lot of positive images (images of faces) and negative images (images without faces) to train the classifiers, then we need to extract the features from it.For this we make use of edge feature and line feature.


**Fig. Line and Edge Features**

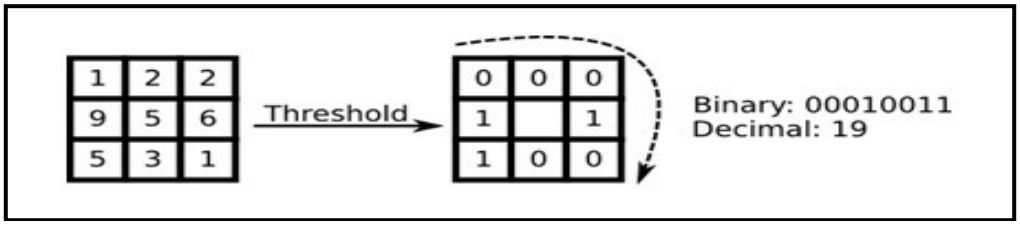
**Edge feature**: It comprises of one black and one white rectangle use to detect edge feature effectively.
**Line feature**: It comprises of one black rectangle between two white rectangles or vice versa used to detect line feature effectively.




The photo at the left suggests an appropriate haar characteristic pixal intensities wherein white is 0 and black is one. The photo at the proper suggests the actual values detected at the photo and its fee stages from zero to 255 and it's far scales right all the way down to zero to at least one for convinence. Delta is the distinction among the common of pixel intensities of mild and darkish location. For a really perfect haar characteristic delta fee is 1. The delta fee for actual haar characteristic is calculated the usage of the above method and examine with an appropriate delta fee. If the actual delta fee is near best delta fee then it's far taken into consideration as haar characteristic.

### 4.2 LBPH

Local binary sample (LBP) is an texture operator which lables the pixel of an photo through threshoolding the community of every pixel and consideres the end result because the binary variety.The primary concept of nearby binary sample is to summerise the nearby shape in an photo through evaluating every pixel with its community.Take a pixel as middle and threshold its acquaintances against. If the depth of the middle pixel is greater-identical its neighbor, then denote it with 1 and zero if not. You'll grow to be with a binary variety for every pixel.So with eight surrounding pixels you may grow to be with 2^eight feasible combinations, known as nearby Binary Patterns or every so often known as LBP codes which might be in decimal values. The first LBP operator defined in literature truly used a set three x three community like this. Here we get decimal/LBP fee for a 3x3 a part of a photo that may lie from zero-255.After the era of LBP fee, histogram of the location is created through counting the variety of comparable LBP values in that location.



Delta is the difference between the average of pixel intensities of light and dark region. For an ideal haar feature delta value is 1. The delta value for real haar feature is calculated using the above formula and compare with the ideal delta value. If the real delta value is close to ideal delta value then it is considered as haar feature.

$$\Delta = \text{dark} - \text{white} = \frac{1}{n}\sum_{\text{dark}}^{n} I(x) - \frac{1}{n}\sum_{\text{white}}^{n} I(x)$$

**Advantages**

The LBP operator is strong towards monotonic grey scale transformations. LBPH can understand each aspect and the front faces. In LBPH every photograph is analyzed independently, at the same time as different algorithms like eigenfaces and fisherfaces approach seems on the dataset as a whole.

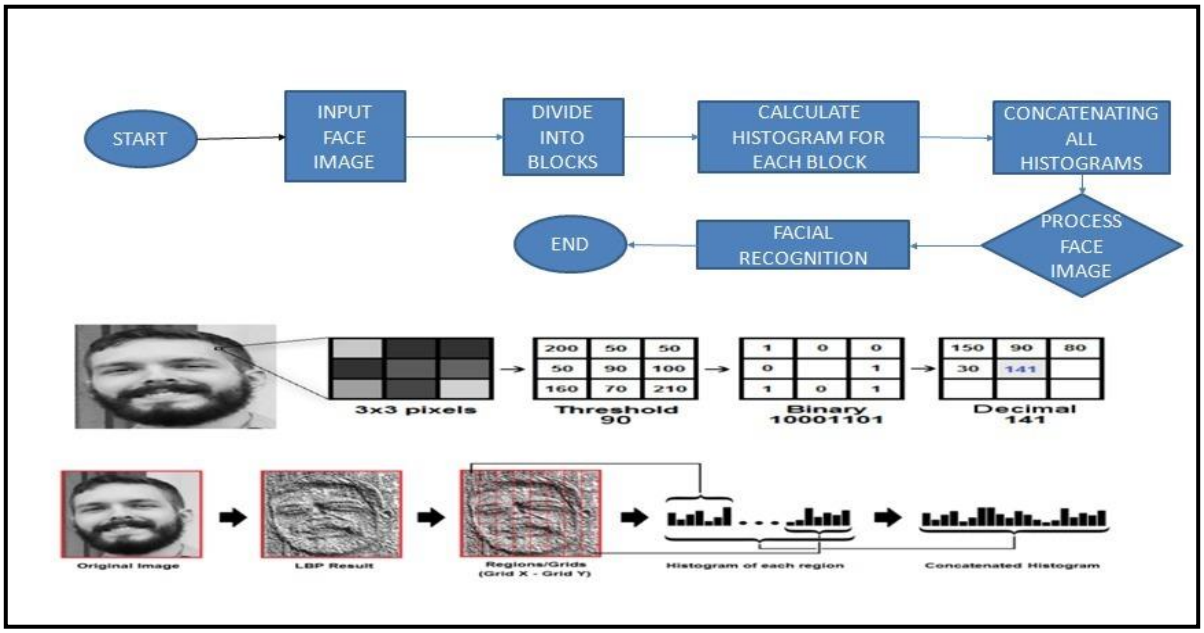



**Fig. Flowchart of LBPH working**

**4.3 Twilio SMS Service:** Twilio's Programmable SMS API helps in hearty informing capacities to your applications. Utilizing this, you can send and get SMS messages, track the conveyance of sent messages, and recover and change message history. Here for the SMS ready we require

- FROM NUMBER-naturally created ph.no utilizing TWILIO account
- TO NUMBER-Registered ph.no
- SSID – Security identifier, is a one of a kind number used to recognize the client

**AUTHENTICATION TOKEN**

With token approval, an optional assistance checks a worker demand. At the point when check is finished, the worker gives a token and reacts to the solicitation. for BOLT/Pi to interface with Twillio accnt for sending sms alarms to enrolled telephone numbers.

**4.4 Visualizing Data on BOLT Cloud**

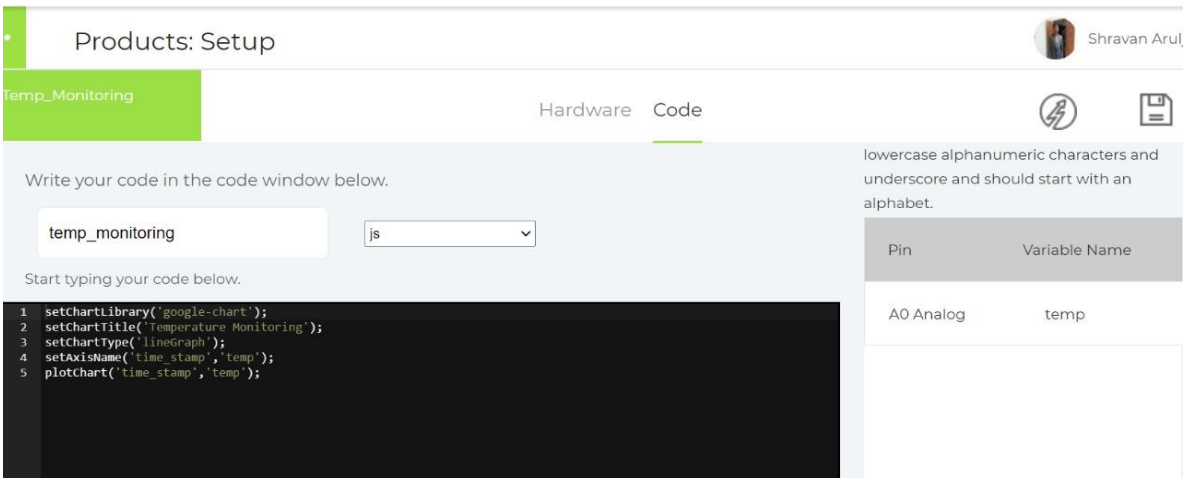



**Fig. Code for Temperature Graph**

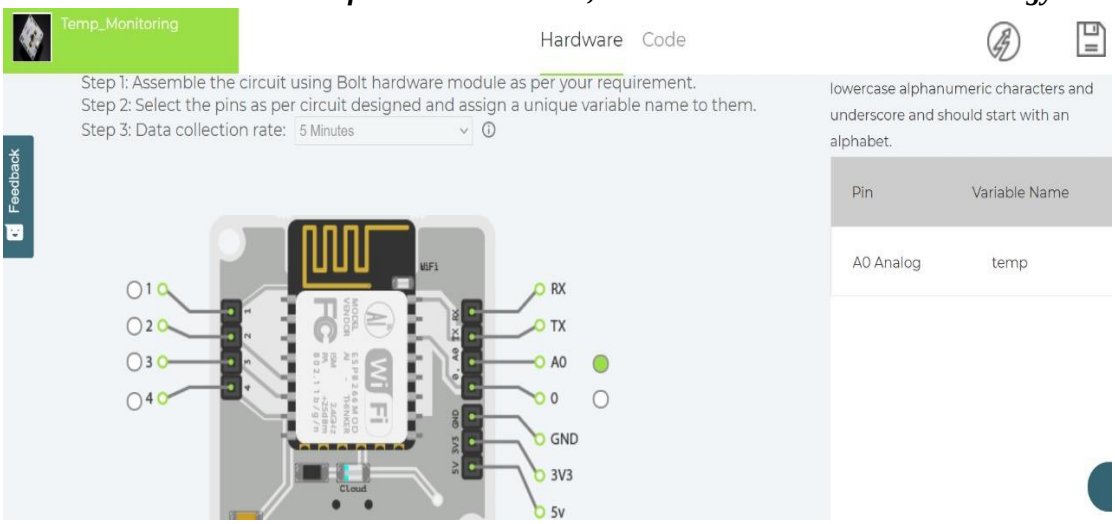



**Fig. Hardware Configuration**

## 5. PROGRAMS

### Programs For Face Detection & Recognition

To solve the problem of theft and accidents that occur at home using a smart security robot. The robot works on raspberry PI operating system which is loaded with thony python IDE. Our python program consists of facial detection program using Open CV package which works on haar cascade classifier for facial detection and recognition.

### 5.1 Face Dataset

This program consists of codes that are used in collection of images of known users and storing it in an database for example a folder or a cloud storage using ***HAAR CASCADE CLASSIFIER.*** When the program runs it takes specified amount of pictures. The output of this program is to store the users faces in database.

```
import cv2
import os
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video width
cam.set(4, 480) # set video height
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# For each person, enter one numeric face id
face_id = input('\n enter user id')
print("\n [INFO] Initializing face capture....")
# Initialize individual sampling face count
count = 0
while(True):
ret, img = cam.read()
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_detector.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
count += 1
# Save the captured image into the datasets folder
cv2.imwrite("dataset/User." + str(face_id) + '.' + str(count) + ".jpg", gray[y:y+h,x:x+w])
cv2.imshow('image', img)
 k = cv2.waitKey(100) & 0xff
if k == 27:
break
elif count >= 30:
break
print("\n [INFO] Exiting Program")
cam.release()
```

### 5.2 FACE TRAINING

We train the stored database images and then these trained images are stored in the ***trainer.yml*** file. The output of this program is to trainer the images of the users stored in the database.

```
import cv2
import numpy as np
from PIL import Image
import os
# Path for face image database
path = 'dataset'
```

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");
# function to get the images and label data
def getImagesAndLabels(path):
    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []
    for imagePath in imagePaths:
        PIL_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_numpy = np.array(PIL_img,'uint8')
        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_numpy)
        for (x,y,w,h) in faces:
            faceSamples.append(img_numpy[y:y+h,x:x+w])
            ids.append(id)
    return faceSamples,ids

print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")
faces,ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(ids))

# Save the model into trainer/trainer.yml
recognizer.write('trainer/trainer.yml')

# Print the numer of faces trained and end program
print("\n [INFO] {0} faces trained.".format(len(np.unique(ids))))
```

**5.3 FACE RECOGNITION**

We are reading the trained images stored in the ***trainer.yml file*** and using ***HAAR cascade classifier*** to classify the faces of users and obtain the accuracies. If there is no set of images in the database then the program shows an error. The output of this program is to recognize multiple faces stored in the database and give their accuracies and if there is a strangers face it shows ***negative accuracy*** and shows unknown.

```
import cv2
import numpy as np
import os
#For SMS Alert
from twilio.rest import Client
t_s = "AC194bfe125d1ad52653f6f5bfd6c18075" #SSID
t_a = "1615bffeb7a0c511b16683609042b4b5"   #AUTH_TOKEN
client = Client(t_s,t_a)
FROM= "+16195971089"  #Trial twillio account number
TO= "+919740798260", "+919036428906", "+919611873279"  #must be registered number on Twillio
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainer/trainer.yml')
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath);
font = cv2.FONT_HERSHEY_TRIPLEX
#iniciate id counter
id = 0
names = [0, 1, 2, 3, 'Z', 'W']
# Initialize and start realtime video capture
cam = cv2.VideoCapture(0)
cam.set(3, 640) # set video widht
cam.set(4, 480) # set video height
# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)
while True:
 ret, img =cam.read()
 gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
 faces = faceCascade.detectMultiScale( gray,
scaleFactor = 1.2,
minNeighbors = 5,
minSize = (int(minW), int(minH)),
        )
 for(x,y,w,h) in faces:
```

```
cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)
 id, confidence = recognizer.predict(gray[y:y+h,x:x+w])
# Check if confidence is less then 100 ==> "0" is perfect match
 if (confidence < 100):
 id = names[id]
confidence = "  {0}%".format(round(100 - confidence))
 else:
id = "unknown"
confidence = "  {0}%".format(round(100 - confidence))
cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255), 2)
cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)
cv2.imshow('camera',img)
k = cv2.waitKey(10) & 0xff
if k == 27:
break
print("\n [INFO] Exiting Program")
cam.release()
cv2.destroyAllWindows()
```

**ARDUINO CODE FOR ROBOT NAVIGATION THROUGH APP(BLUETOOTH RC CONTROLLER):**

```
#define CON_MOTOR1 0
#define CON_MOTOR2 0
// Motor shield uses four pins - 4, 5, 6, 7 to manage the motors
// 4 and 7 — for direction, 5 and 6 — for speed
#define SPEED_1 5
#define DIR_1 4
#define SPEED_2 6
#define DIR_2 7
// Shortcuts for possible robot movements
#define FORWARD 0
#define BACKWARD 1
#define LEFT 2
#define RIGHT 3
// Variable for bluetooth buffer
int RX_buff;
/* * This function is used to manage actual movements */
void go (int newDirection, int speed) {
boolean motorDirection_1, motorDirection_2;
switch ( newDirection ) {
case FORWARD: motorDirection_1 = true; motorDirection_2 = true; break;
case BACKWARD: motorDirection_1 = false; motorDirection_2 = false; break;
case LEFT: motorDirection_1 = true; motorDirection_2 = false; break;
case RIGHT: motorDirection_1 = false; motorDirection_2 = true; break;
}
// In case of motor set-up mistakes just change the numbers
motorDirection_1 = CON_MOTOR1 ^ motorDirection_1;
motorDirection_2 = CON_MOTOR2 ^ motorDirection_2;
// Let's move!
analogWrite(SPEED_1, speed);
analogWrite(SPEED_2, speed);
digitalWrite(DIR_1, motorDirection_1);
digitalWrite(DIR_2, motorDirection_2);
}
void setup() {
Serial.begin(9600);
// Sets pins 4, 5, 6, 7 to output mode
for(int i = 4; i <8; i++)
pinMode(i, OUTPUT);

delay(5000);
}
void loop() {
// Reading bluetooth data
RX_buff = Serial.read();
// Responding to bluetooth data
if (RX_buff == 1) {
```

```
go(FORWARD, 125);
delay(1000);
}
if (RX_buff == 2){
go(LEFT, 80);
delay(1500);
}
if (RX_buff == 4){
go(BACKWARD, 70);
delay(1500);
}
if (RX_buff == 3)
{
go(RIGHT, 80);
delay(1500);
}
// Stop,if necessary
if (RX_buff == 5 || RX_buff == 0){
analogWrite(SPEED_1, 0);
analogWrite(SPEED_2, 0);
}
}
```

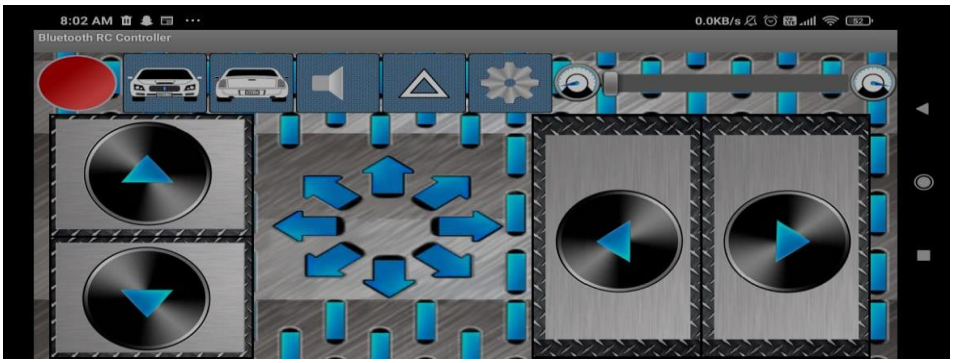



**Fig. Bluetooth RC Controller App**

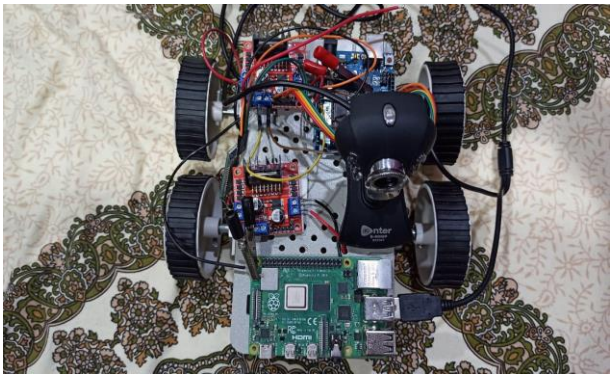


**Fig. Developed Robot**

## 6. RESULTS

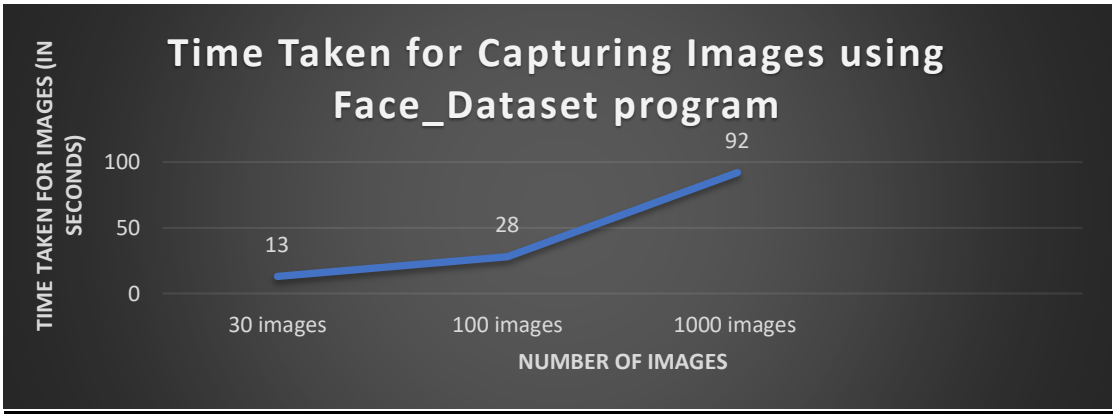



**Figure. Time taken (seconds) vs Number of Images to be stored in dataset**

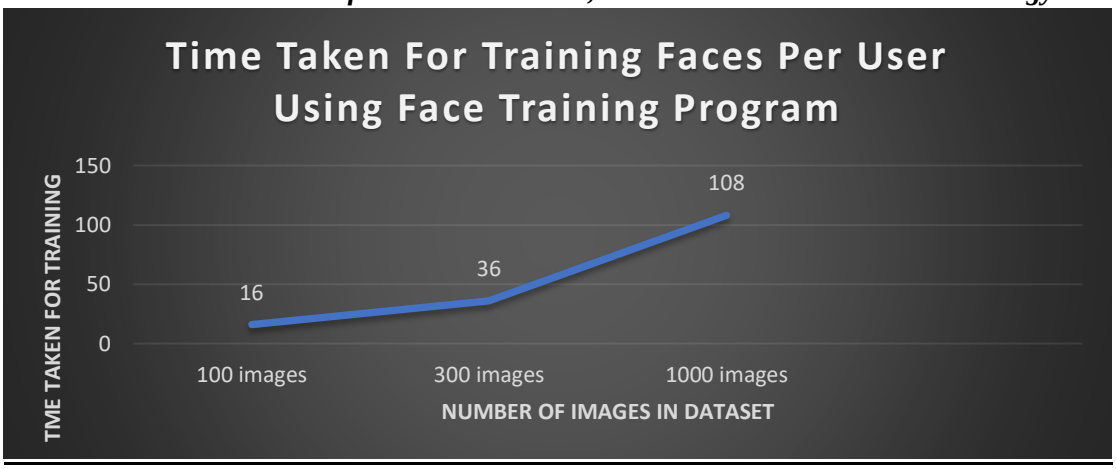



**Figure. Time taken vs Number of Images Trained**

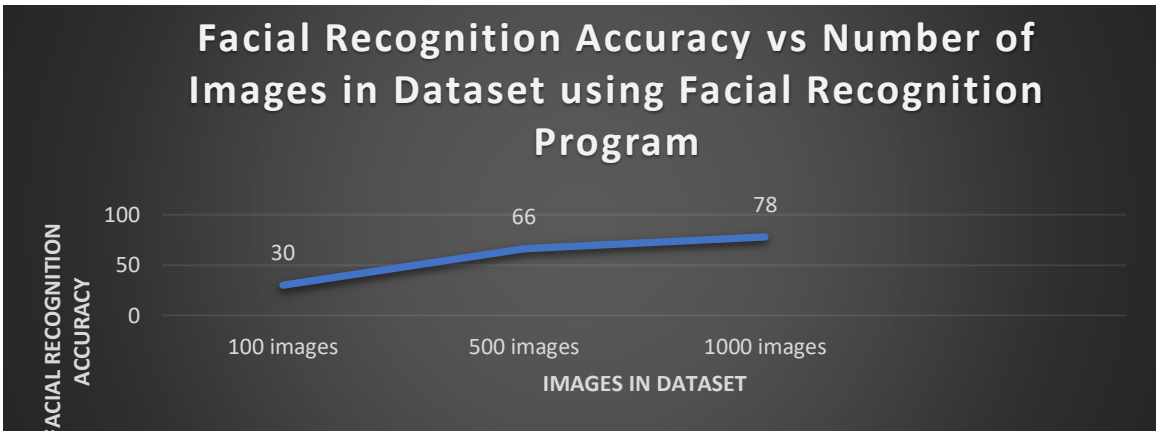



**Figure. Facial Recognition Accuracy vs Number of Images in Dataset**

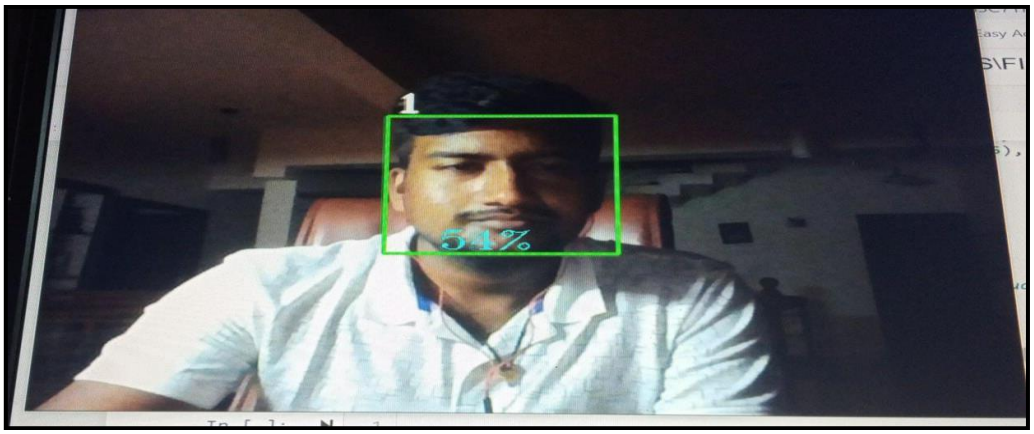



**Fig. Facial Recognition Accuracy Obtained using 30 images on Raspberry Pi 3**

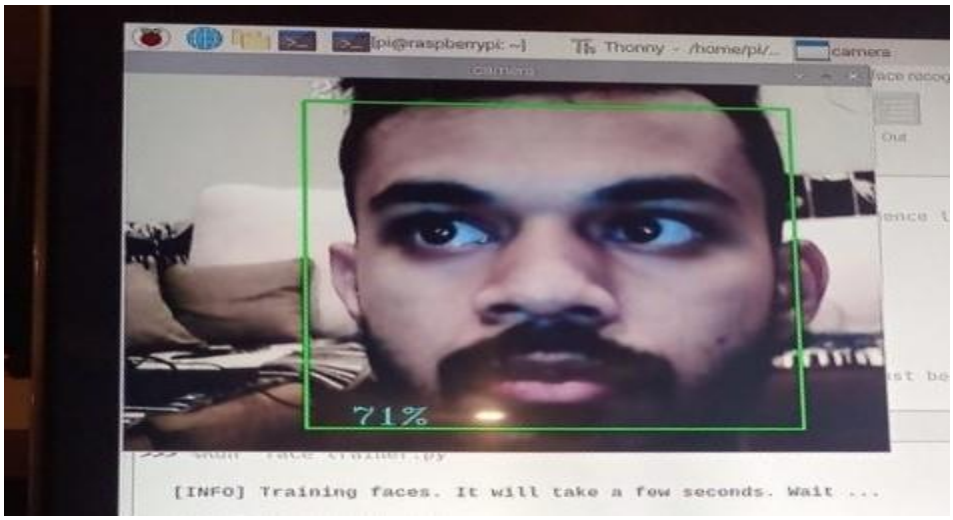



**Fig. Facial Recognition Accuracy Obtained using 100 images on Raspberry Pi 4**

**6.1 Facial Recognition Accuracy Calculation using LBPH**

Using LBPH algorithm, the same process is followed for the new face detected by HaarCascade. After generating the histogram for the new face, it is compared with histograms for each user.
This comparison is done by Euclidean distance formula.

The Euclidean distance is calculated by comparing the new image features with features stored in the dataset. The algorithm will return the ID number (which was given as input using the "Face Dataset" program) as an output from the image with the closest histogram. The algorithm will also return the calculated called confidence measurement.

$$D = \sqrt{\sum_{i=1}^{n} \left(hist1_i - hist2_i\right)^2}$$

Where hist1i and hist2i are the histograms of the images to compare and D is the value of the Euclidean distance. If D is small, face is recognized accurately.

**6.2 Temperature Data Visualization on BOLT Cloud**

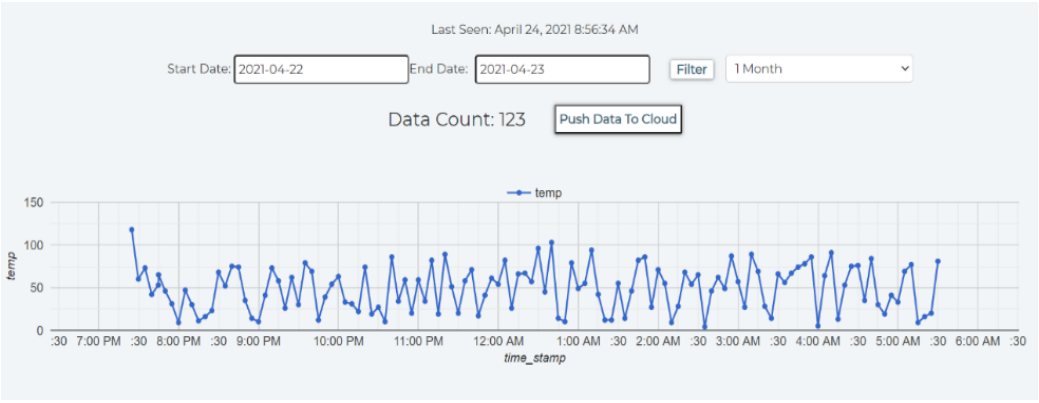



**Fig. Temperature Graph on BOLT Cloud**

This graph shows Temperature (y-axis) vs Time Stamp (x-axis) for 1 day. LM35 temperature sensor gives output data in form of constant values on BOLT Cloud. Temperature=(100*sensor_value)/1024 (formula to convert the sensor analog values to Celsius value).

**6.3 SMS Alert using Twilio Messaging Service**

User receives alerts to his phone under any of the 2 cases.
Case 1-Intruder Alert: If a face is detected but not recognized using the python programs
Case 2-Temperature Alert: If the LM35 sensor senses a temperature value outside the safe range.

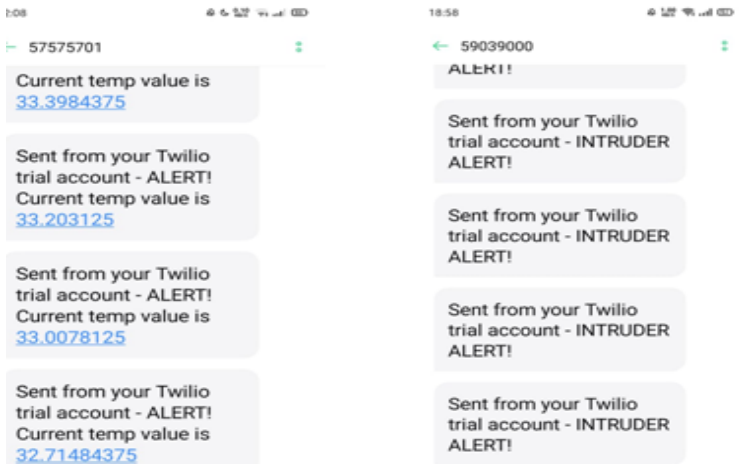



**Fig. Temperature Alert and Intruder Alert using Twilio SMS service**

**7. CONCLUSION**

- The accuracy for facial recognition is high which shows that LBPH and Haar Cascade Classifier are efficient and are compatible with each other.
- The time taken for taking images, training them & facial recognition accuracy depend on the lighting conditions and quality of webcam used. The results have been found using 5MP 1080p recording Logitech webcam.
- The faces can be detected and recognized within 100 cm.
- The accuracy of facial recognition depends on the number of images of the owner stored in the database using the Face Dataset Program. The time required to run the programs depend on the version of Raspberry Pi and the internet connection on Raspberry Pi. The accuracy in Raspberry Pi 3 is 68% and version 4 has 80% with same number of images (100 images) in face dataset.

- For better recognition we can use better quality webcam, bright light conditions and a greater number of images in face dataset.
- Future scope involves autonomous navigation for robots, using night vision camera and neural networks for face detection & recognition.
-

**REFERENCES**

[1] W.Han Hung, Peter Liu, and Shih-Chung Kang, "Service-Based Simulator for Security Robot, Advanced robotics and Its Social Impacts, 2008. ARSO 2008. IEEE Workshop on.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73. J.

[2] Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] Viswaprakash Babu & Vikram Kulkarni,"Embedded smart car security system on face detection', special issue of IJCCT,ISSN(Online):2231-0371,ISSN(Print): 09757449, volume-3, issue-1.

[4] Asaad.M.J.AlHindawi,IbraheemTalib,"Experimentally Evaluation of GPS/GSM Based SystemDesign", Journal of Electronic Systems Volume 2 Number 2 June 2012.

[5] Frazer.K.Noble,"Comparison of OpenCV's feature detectors and feature matchers," *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*,IEEE, 2016, pp. 1-6, doi: 10.1109/M2VIP.2016.7827292.

[6] Abhishek Pratap Singh," Face Recognition System Based on LBPH Algorithm", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-8, Issue-5S, May, 2019.

[7] H. Abdulsamet and T. Olcay, "Identification system from motion pictures: LBPH application," *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017, pp. 845-850, doi: 10.1109/UBMK.2017.8093546.

**BIOGRAPHY**

**Shravan Aruljothi**
**New Horizon College of Engineering, Bengaluru, Karnataka, India**

**Sharad Dewanand Parate**
**New Horizon College of Engineering, Bengaluru, Karnataka, India**

**Harshit S.**
**New Horizon College of Engineering, Bengaluru, Karnataka, India**

**Nehal Andani**
**New Horizon College of Engineering, Bengaluru, Karnataka, India**