A PROJECT REPORT

ON

**SMART SECURITY SYSTEM FOR HOMES**

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS OF THE DEGREE OF

B. Tech

BY

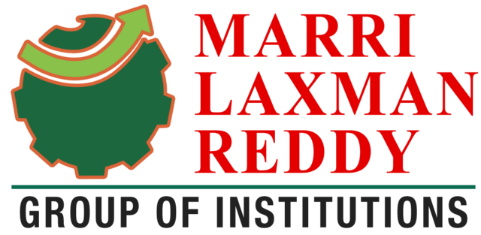
P. VAMSHI KRISHNA 187Y5A0451

SHIRISHA 187Y1A04A8

SAI VARDHAN 187Y1A04F9

UNDER THE ESTEEMED GUIDANCE OF





Marri Laxman Reddy Institute of Technology and Management

2019

ACKNOWLEDGEMENT

A dissertation work, which integrated skill full and hard work need a great deal of guidance and helping hand. I would like to add a few heartfelt words for the people who have directly or indirectly in preparing in this dissertation work in numerous ways.

My heartfelt thanks to **Marri Laxman Reddy**, chairman of our college for providing me an opportunity to care out this project work form this college.

My sincere thanks to **Dr. K Venkateswara Reddy**, the principal of our college for providing me the required facilities to complete my project.

With profound sense of gratitude and regards, I acknowledge with great pleasure the Guidance and support extended by, **Mr. K Abdul Basith& Mr. K Nagabhushanam**, heads of the Computer Science &Electronics and Communication Engineeringdepartments and internal guide, **Mr. B Prasad**for his timely suggestion, logical thought that have greatly contributed in bringing out the project in a systematic way. He has been inculcating innovative skills and technicalities at all stages at performing this project.

My sincere thanks to all the teaching and non- teaching staff of our department for their constant co-operation extended to me to complete the project successfully. My thanks to all my classmates and friends for their constant help and support during my project work.

My heartfelt gratitude to my parents and my family for their constant encouragement throughout the work carried out to prepare this project report.

SUBMITTED BY

BATCH NO- 09

P .VAMSHI KRISHNA 187Y5A0451

SHIRISHA 187Y1A04A8

SAI VARDHAN 187Y1A04F9

**INDEX**

Introduction………………………………………………………….. 6-7

Block Diagram……………………………………………………….. 8

Hardware

* Raspberry Pi3……………………………………………… 9-10
* LogiTech Camera………………………………………….. 11-12
* Servo Motor………………………………………………... 13
* Connecting wires………………………………………….. 14

Software

* Python IDLE……………………………………………… 15-17
* OpenCV ………………………………………………….. 17-21

Description of the project…………………………………………… 22-30

Advantages of smart security system ……………………………… 31

Conclusion…………………………………………………………… 32

**ABSTRACT**

Now a days the technology is increasing rapidly,tha leads to an upgradation in home security system.Automation in security sector makes it more authentic.There are many electrical equipment’s are available in home which are in necessity of monitoring from a remote area all at a time.In paper a home security system is proposed along with the face detection technique.Astand alone system through Internet of Things as a network of communication is implemented.Raspberry Pi is use as controlling unit codded in Python language.

Home Security Systems are a need of the modern day houses. It is possible to design a simple home security solution by using Raspberry Pi and utilizing the power of Internet of Things. The home security system designed in this project is a simple and easily installable device built using Raspberry Pi 3, Web Cam and servo motor.

The device designed in this project can be installed at the main entrance of a house. It detects motion of any visitor and starts capturing the images with the help of a USB web cam. The images are temporarily stored on the Raspberry Pi.. . The Pi uses OpenCV library to capture images from the Web Cam and saves the images in a specicified folder.

The home security system designed in this project, though being simple, is a powerful application. The user can keep surveillance of his house from anywhere, any time and always by just installing this small device at the main entrance. Many such devices can also be installed to further add security layers.

**INTRODUCTION**

The present scenario ensures the safety and security has become an inevitably essential. There is a regressive progress in the security system as the influence of modern technology is reaching its peak. When there is a modern home with minimum human effort, it’s well known as modern home. Since there is an advent of wireless and digital technologies, all together it introduces a automated intelligent security system. The automated home security system can be designed with the surveillance camera .



The new IoT based products and services will grow exponentially in next few years predicted by the analysts. The IoT involves different link layer technologies and a huge range of devices. IoT provides open access to particular set of data. Raspberry Pi is a compact minicomputer which is smart enough to give the good connectivity to theinternet as well as boost up the signals. This framework mainly envelopes the home security system from the sensor, networking, integrates real time data and data management. This proposed system has high latency and low cost. The system is highly reliable and consumes very less power in comparison with existing system. The home security system based on some camera connected to the home and the output for this is in real time with the minimum delay in the operation.

The objective of this paper is home security using Raspberry Pi . Images of authorized person is stored in the data base and when some human encounter the camera, camera will capture the image and compare that with the data base. When the image matches with the data base the name pop us will come of that particular authorized human and opens the door.



BLOCK DIAGRAM:

PYTHON IDLE

RASPBERRY PI3

LOGITECH CAMERA

SERVO MOTOR

OPENCV

Hardware

**The Hardware that I used in this project**:

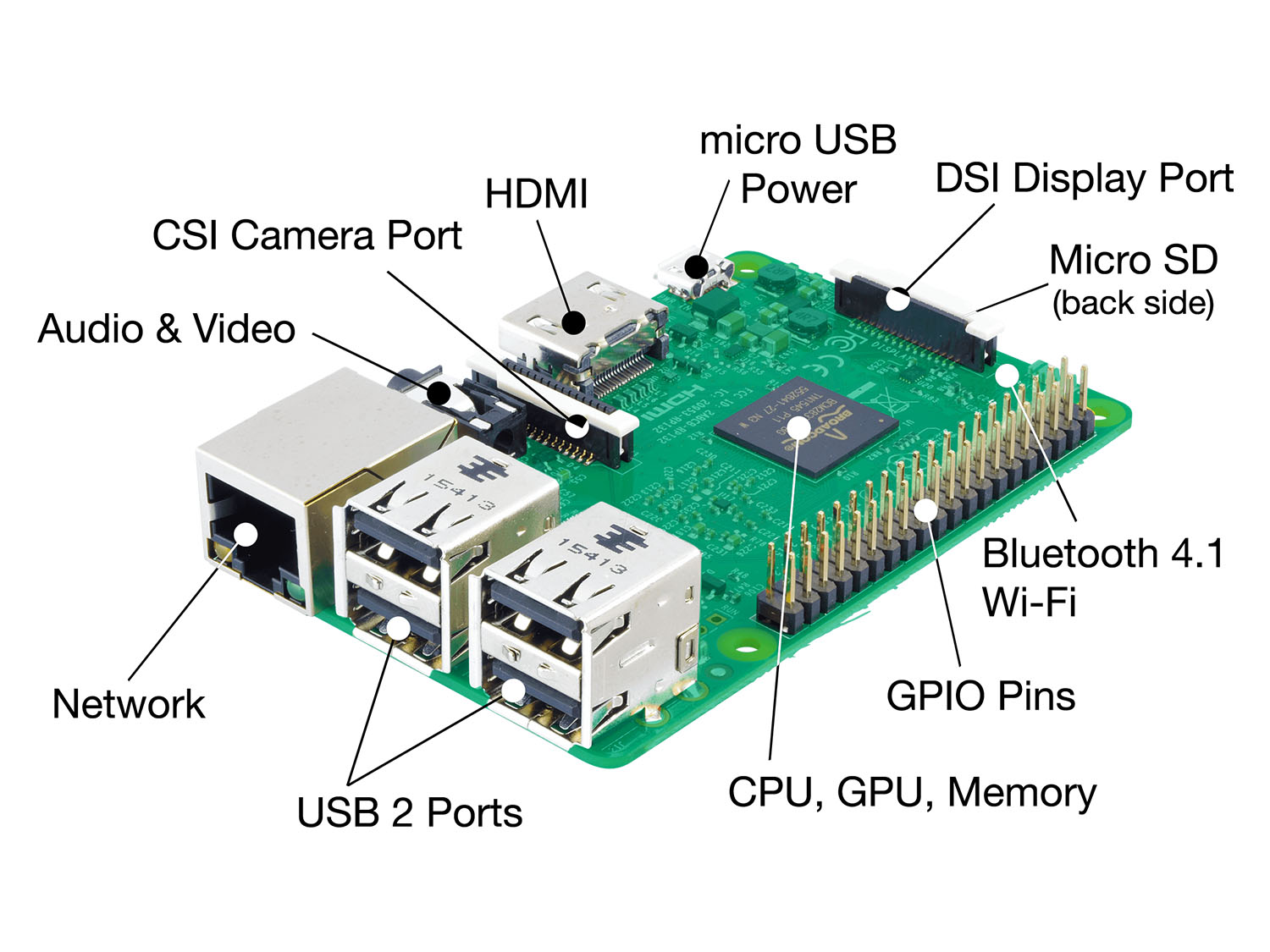
* *Raspberry Pi3*
* *LogiTech Camera*
* *Servo Motor*
* *Connecting wires*

**Raspberry Pi3:**

Raspberry Pi 3 is the third generation Raspberry Pi. It is a miniature marvel, packing considerable computing power into a footprint no larger than a credit card. The processor at the heart of the Raspberry Pi system is a Broadcom BCM2837 system-on-chip (SoC) which houses a 1.2 GHz Quad Core ARM Cortex-A53 processor. The vast majority of the system’s components, including its central and graphics processing units along with the audio and communications hardware, are built onto that single component along with 1 GB LPDDR2 memory chip at the centre of the board. It is not just this SoC design that makes the BCM2837 different to the processor found in a typical desktop or laptop, however, it also uses a different instruction set architecture (ISA), known as ARM.

The Pi comes equipped with on-board 10/100 BaseT Ethernet Socket, HDMI and Composite RCA port for video, 3.5 mm audio output jack, 15-pin MIPI Camera Serial Interface (CSI-2), Display Serial Interface, Bluetooth 4.1, 802.11 b/g/n Wireless LAN, Micro SDIO for Micro SD Card, 4 USB 2.0 Connectors, 40 pin header containing 27 GPIO pins and Micro USB socket for power supply.

The Raspberry Pi is a single board computer and is designed to run an operating system called GNU/Linux Raspbian. Hereafter referred to simply as Linux. Unlike Windows or OS X, Linux is open source, so it is possible to download the source code for the entire operating system and make whatever changes desired. The Raspberry Pi 3 can also run Windows 10 IoT and many other embedded operating systems most of which are Linux derivatives. The operating system should be loaded in a MicroSD card and boot from it. With powerful computing resources, large number of multimedia interfaces and GPIO  pins, Raspberry Pi 3 is a suitable choice to run a software oriented complex IoT or Embedded project that requires sufficient computing power as well as large scale sensor connectivity. With on-board Bluetooth and Wi-Fi, this 3rd generation Pi can be easily deployed in an IoT network



**Logitech Camera:**

Logitech has released its new [Logitech BRIO 4K Pro Webcam](http://www.logitech.com/en-us/product/4k-pro-webcam) with support for Windows Hello and other secure infrared-based facial recognition applications.

The webcam is equipped with a second infrared LED and sensor, enabling users to combine it with the primary sensor and configured software to make the device Windows Hello certified so it can be used with the Microsoft security feature to login via facial recognition.



Logitech Webcam Software is a free software that allows you to use your Logitech webcam with your desktop computer. With software that supports a number of different Logitech webcam models, get the most out of your camera and connect with others using high-quality video and images. Logitech Webcam Software bridges the game between two different pieces of hardware and creates a simple and easy-to-use recording experience.

Face Tracker recognizes when a person is sitting in front of the camera, zones in on the face and begins automatically tracking. Normal Webcams have a 10-degree field that the face will fit in before drifting out of the picture. Evans says his company's digital technology (there's no moving parts to adjust the camera position) expands the camera's perspective to about 60 degrees.



**Servo Motor:**

There are lots of servo motors available in the market and each one has its own speciality and applications. The following two paragraphs will help you identify the right type of servo motor for your project/system.

Most of the hobby Servo motors operates from 4.8V to 6.5V, the higher the voltage higher the torque we can achieve, but most commonly they are operated at +5V.  Almost all hobby servo motors can rotate only from 0° to 180° due to their gear arrangement so make sure you project can live with the half circle if no, you can prefer for a 0° to 360° motor or modify the motor to make a full circle. The gears in the motors are easily subjected to wear and tear, so if your application requires stronger and long running motors you can go with metal gears or just stick with normal plastic gear.



Next comes the most important parameter, which is the **torque** at which the motor operates. Again there are many choices here but the commonly available one is the 2.5kg/cm torque which comes with the Towerpro SG90 Motor. This 2.5kg/cm torque means that the motor can pull a weight of 2.5kg when it is suspended at a distance of 1cm. So if you suspend the load at 0.5cm then the motor can pull a load of 5kg similarly if you suspend the load at 2cm then can pull only 1.25. Based on the load which you use in the project you can select the motor with proper torque. The below picture will illustrate the same.

After selecting the right Servo motor for the project, comes the question how to use it. As we know there are three wires coming out of this motor. The description of the same is given on top of this page. To make this motor rotate, we have to power the motor with +5V using the Red and Brown wire and send PWM signals to the Orange colour wire. Hence we need something that could generate PWM signals to make this motor work, this something could be anything like a 555 Timer or other Microcontroller platforms like Arduino, PIC, ARM or even a microprocessor like Raspberry Pie.

**Connecting wires:**

A **jump wire** (also known as jumper wire, or jumper) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.



**Software:**

* *Python IDLE*
* *OpenCV*

**Python IDLE:**

IDLE is Python’s Integrated Development and Learning Environment.

The features of IDLE are :It is coded in 100% pure Python, using the tkinter GUI toolkit

cross-platform: works mostly the same on Windows, Unix, and macOS

Python shell window (interactive interpreter) with colorizing of code input, output, and error messages

Multi-window text editor with multiple undo, Python colorizing, smart indent, call tips, auto completion, and other features

Search within any window, replace within editor windows, and search through multiple files (grep)

Debugger with persistent breakpoints, stepping, and viewing of global and local namespaces

configuration, browsers, and other dialogs.

IDLE has two main window types, the Shell window and the Editor window. It is possible to have multiple editor windows simultaneously. On Windows and Linux, each has its own top menu. Each menu documented below indicates which window type it is associated with.

Output windows, such as used for Edit => Find in Files, are a subtype of editor window. They currently have the same top menu but a different default title and context menu.

On macOS, there is one application menu. It dynamically changes according to the window currently selected. It has an IDLE menu, and some entries described below are moved around to conform to Apple guidelines.

**Python Shell window:**

With IDLE’s Shell, one enters, edits, and recalls complete statements. Most consoles and terminals only work with a single physical line at a time.

When one pastes code into Shell, it is not compiled and possibly executed until one hits return. One may edit pasted code first. If one pastes more that one statement into Shell, the result will be a syntaxerror when multiple statements are compiled as if they were one.

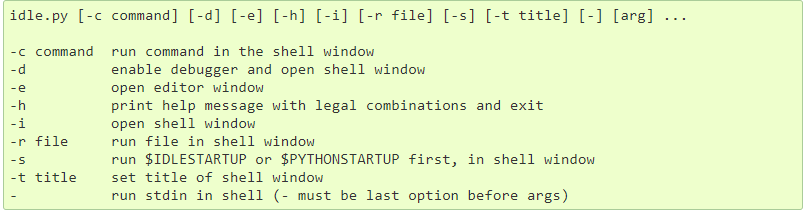
The editing features described in previous subsections work when entering code interactively. IDLE’s Shell window also responds to the following keys.

* C-c interrupts executing command
* C-d sends end-of-file; closes window if typed at a >>> prompt
* Alt-/ (Expand word) is also useful to reduce typing

Command history

* Alt-p retrieves previous command matching what you have typed. On macOS use C-p.
* Alt-n retrieves next. On macOS use C-n.
* Return while on any previous command retrieves that command

**Command line usage:**



If there are arguments:

* If -, -c, or r is used, all arguments are placed in sys.argv[1:...] and sys.argv[0] is set to '', '-c', or '-r'. No editor window is opened, even if that is the default set in the Options dialog.
* Otherwise, arguments are files opened for editing and sys.argv reflects the arguments passed to IDLE itself.

**OpenCV:**

**OpenCV** (*Open source computer vision*) is a library of programming functions mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision).[[1]](https://en.wikipedia.org/wiki/OpenCV#cite_note-1) Originally developed by [Intel](https://en.wikipedia.org/wiki/Intel_Corporation), it was later supported by [Willow Garage](https://en.wikipedia.org/wiki/Willow_Garage) then Itseez (which was later acquired by Intel[[2]](https://en.wikipedia.org/wiki/OpenCV#cite_note-2)). The library is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source_software) [BSD license](https://en.wikipedia.org/wiki/BSD_license).

OpenCV supports the [deep learning](https://en.wikipedia.org/wiki/Deep_learning) frameworks [TensorFlow](https://en.wikipedia.org/wiki/TensorFlow), [Torch](https://en.wikipedia.org/wiki/Torch_(machine_learning))/[PyTorch](https://en.wikipedia.org/wiki/PyTorch) and [Caffe](https://en.wikipedia.org/wiki/Caffe_(software)).

OpenCV's application areas include:

* 2D and 3D feature toolkits
* [Facial recognition system](https://en.wikipedia.org/wiki/Facial_recognition_system)
* [Gesture recognition](https://en.wikipedia.org/wiki/Gesture_recognition)
* [Human–computer interaction](https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction) (HCI)
* [Mobile robotics](https://en.wikipedia.org/wiki/Mobile_robotics)
* Motion understanding
* Object identification
* [Segmentation](https://en.wikipedia.org/wiki/Segmentation_(image_processing)) and recognition
* [Structure from motion](https://en.wikipedia.org/wiki/Structure_from_motion) (SFM)
* [Motion tracking](https://en.wikipedia.org/wiki/Video_tracking)
* [Augmented reality](https://en.wikipedia.org/wiki/Augmented_reality)

**Programming language for OpenCV:**

OpenCV is written in [C++](https://en.wikipedia.org/wiki/C%2B%2B) and its primary interface is in C++, but it still retains a less comprehensive though extensive older [C interface](https://en.wikipedia.org/wiki/C_(programming_language)). There are bindings in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and [MATLAB](https://en.wikipedia.org/wiki/MATLAB)/[OCTAVE](https://en.wikipedia.org/wiki/GNU_Octave). The API for these interfaces can be found in the online documentation.[[12]](https://en.wikipedia.org/wiki/OpenCV#cite_note-Cdocs-12) Wrappers in other languages such as [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Perl](https://en.wikipedia.org/wiki/Perl),[[13]](https://en.wikipedia.org/wiki/OpenCV#cite_note-13) [Ch](https://en.wikipedia.org/wiki/Ch_(computer_programming)),[[14]](https://en.wikipedia.org/wiki/OpenCV#cite_note-14) [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)),[[15]](https://en.wikipedia.org/wiki/OpenCV#cite_note-15) and [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)) have been developed to encourage adoption by a wider audience.

Since version 3.4, **OpenCV.js** is a [JavaScript](https://en.wikipedia.org/wiki/JavaScript) binding for selected subset of OpenCV functions for the web platform. [[16]](https://en.wikipedia.org/wiki/OpenCV#cite_note-16)

All of the new developments and algorithms in OpenCV are now developed in the C++ interface.

**Hardware acceleration:**

If the library finds Intel's [Integrated Performance Primitives](https://en.wikipedia.org/wiki/Integrated_Performance_Primitives) on the system, it will use these proprietary optimized routines to accelerate itself.

A [CUDA](https://en.wikipedia.org/wiki/CUDA)-based [GPU](https://en.wikipedia.org/wiki/Graphics_processing_unit) interface has been in progress since September 2010.[[17]](https://en.wikipedia.org/wiki/OpenCV#cite_note-OpenCVGPU-17)

An [OpenCL](https://en.wikipedia.org/wiki/OpenCL)-based [GPU](https://en.wikipedia.org/wiki/Graphics_processing_unit) interface has been in progress since October 2012,[[18]](https://en.wikipedia.org/wiki/OpenCV#cite_note-OpenCVOCL-18) documentation for version 2.4.13.3 can be found at docs.opencv.org.

**OS Support:**

OpenCV runs on the following desktop operating systems: [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [Linux](https://en.wikipedia.org/wiki/Linux), [macOS](https://en.wikipedia.org/wiki/MacOS), [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD), [NetBSD](https://en.wikipedia.org/wiki/NetBSD), [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD). OpenCV runs on the following mobile operating systems: [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), [iOS](https://en.wikipedia.org/wiki/IOS), [Maemo](https://en.wikipedia.org/wiki/Maemo),[[20]](https://en.wikipedia.org/wiki/OpenCV#cite_note-Maemo_Port-20) [BlackBerry 10](https://en.wikipedia.org/wiki/BlackBerry_10).[[21]](https://en.wikipedia.org/wiki/OpenCV#cite_note-21) The user can get official releases from [SourceForge](https://en.wikipedia.org/wiki/SourceForge) or take the latest sources from [GitHub](https://en.wikipedia.org/wiki/GitHub).[[22]](https://en.wikipedia.org/wiki/OpenCV#cite_note-22) OpenCV uses [CMake](https://en.wikipedia.org/wiki/CMake).

1)Back ground on face recogniotion:

* **Face Detection:**Look at the picture and find a face in it.
* **Data Gathering:**Extract unique characteristics of face that it can use to differentiate him from another person, like eyes, mouth, nose, etc.
* **Data Comparison:**Despite variations in light or expression, it will compare those unique features to all the features of all the people you know.
* **Face Recognition:**It will determine the person in the photo.

2)Theory of opencv face recognizers:

There are three easy steps to computer coding facial recognition, which are similar to the steps that our brains use for recognizing faces. These steps are:

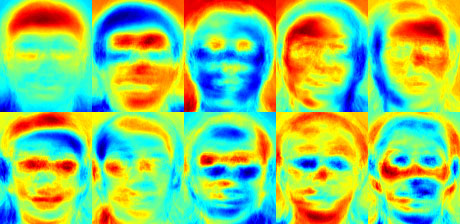
Data Gathering: Gather face data (face images in this case) of the persons you want to identify.

Train the Recognizer: Feed that face data and respective names of each face to the recognizer so that it can learn.

Recognition: Feed new faces of that people and see if the face recognizer you just trained recognizes them.

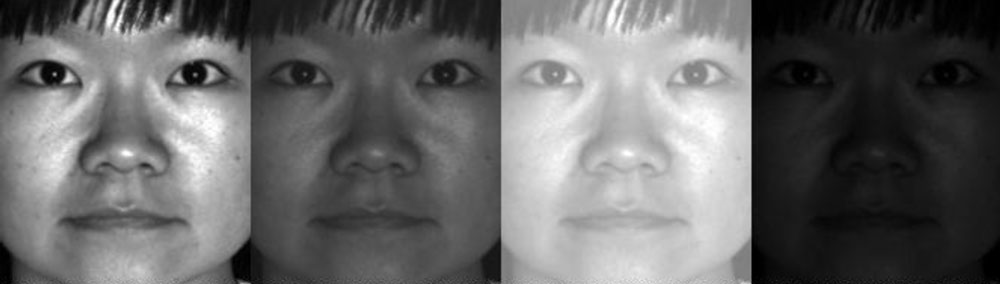
2.1)Eigenfaces face recognizer:

This algorithm considers the fact that **not all parts of a face are equally important or useful for face recognition**. Indeed, when you look at someone, you recognize that person by his distinct features, like the eyes, nose, cheeks or forehead; and how they vary respect to each other.



2.2)Fisherfacesface recognizer:

This algorithm is an improved version of the last one. As we just saw, EigenFaces looks at all the training faces of all the people at once and finds principal components from all of them combined. By doing that, it doesn't focus on the features that discriminate one individual from another. Instead, it concentrates on the ones that represent all the faces of all the people in the training data, as a whole.



Since EigenFaces also finds illumination as a useful component, it will find this variation very relevant for face recognition and may discard the features of the other people's faces, considering them less useful. In the end, the variance that EigenFaces has extracted represents just one individual's facial features.

**2.3)**Local binary patterns histograms (LBPH) Face Recognizer:

I wrote a detailed explanation of Local Binary Patterns Histograms in my previous article on [face detection](https://www.superdatascience.com/blogs/opencv-face-detection/), which I’m sure you’ve read by now. So, here I will just give a brief overview of how it works.

We know that Eigenfaces and Fisherfaces are both affected by light and, in real life, we can't guarantee perfect light conditions. *LBPH face recognizer is an improvement to overcome this drawback.*

The idea with **LBPH** is not to look at the image as a whole, but instead, try to find its local structure by comparing each pixel to the neighboring pixels.

The LBPH Face Recognizer Process:

Take a 3×3 window and move it across one image. At each move (each local part of the picture), compare the pixel at the center, with its surrounding pixels. Denote the neighbors with intensity value less than or equal to the center pixel by 1 and the rest by 0.

After you read these 0/1 values under the 3×3 window in a clockwise order, you will have a binary pattern like 11100011 that is local to a particular area of the picture. When you finish doing this on the whole image, you will have a list of **local binary patterns**.



**3)**Coding Face Recognition using Python and OpenCV

We are going to divide the Face Recognition process in this tutorial into three steps:

1. Prepare Training Data:Read training images for each person/subject along with their labels, detect faces from each image and assign each detected face an integer label of the person it belongs.
2. Train Face Recognizer:Train OpenCV's LBPH recognizer by feeding it the data we prepared in step 1.
3. Prediction:Introduce some test images to face recognizer and see if it predicts them correctly.

**PROCEDURE**

**Software implementation**:

**Steps for Opencv installation:**

Firstly we should download Opencv packages in your computer

This steps are followed to install opencv on rasberrypi3 in terminal

$ sudo apt-get install python2.7-dev python3-dev

$ cd ~

$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip

$ unzip opencv.zip

$ wget -O opencv\_contrib.zip https://github.com/Itseez/opencv\_contrib/archive/3.3.0.zip

$ unzip opencv\_contrib.zip

$ wget https://bootstrap.pypa.io/get-pip.py

$ sudo python get-pip.py

$ sudo python3 get-pip.py

$ pip install numpy

$ cd ~/opencv-3.3.0/

$ mkdir build

$ cd build

$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE \

-D CMAKE\_INSTALL\_PREFIX=/usr/local \

-D INSTALL\_PYTHON\_EXAMPLES=ON \

-D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib-3.3.0/modules \

-D BUILD\_EXAMPLES=ON ..

make -j4

$ sudo make install

$ sudo ldconfig

Open up a new terminal,

>>> import cv2

>>> cv2.\_\_version\_\_

'3.3.0'

**Steps for webcam installation:**

First,install the fswebcam package : sudo apt-get install fswebcam

To take a picture with camera use the command : fswebcam image.jpg

The output will be shown in this below format:

--- Opening /dev/video0...

Trying source module v4l2...

/dev/video0 opened.

No input was specified, using the first.

Adjusting resolution from 384x288 to 352x288.

--- Capturing frame...

Corrupt JPEG data: 2 extraneous bytes before marker 0xd4

Captured frame in 0.00 seconds.

--- Processing captured image...

Writing JPEG image to 'image.jpg'.

To mention or to change the resolution of image we use : fswebcam –r 1280\*720 image2.jpg

The output will be shown in this format:

--- Opening /dev/video0...

Trying source module v4l2...

/dev/video0 opened.

No input was specified, using the first.

--- Capturing frame...

Corrupt JPEG data: 1 extraneous bytes before marker 0xd5

Captured frame in 0.00 seconds.

--- Processing captured image...

Writing JPEG image to 'image2.jpg'.

**Python IDLE:**

We need to write a program to develop any project. Now first step is to write program in Python IDLE. We should create a data of n number pics of database to train the module . The Raspberry Pi3 is connected to the system.

**For creating the data:**

import cv2, sys, numpy, os

haar\_file = 'haarcascade\_frontalface\_default.xml'

datasets = 'datasets'

sub\_data = 'name'

path = os.path.join(datasets, sub\_data)

if not os.path.isdir(path):

os.mkdir(path)

(width, height) = (130, 100)

face\_cascade = cv2.CascadeClassifier(haar\_file)

webcam = cv2.VideoCapture(0)

count = 1

while count < 100:

(\_, im) = webcam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.2, 5)

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

cv2.rectangle(im, (x, y), (x + w, y + h), (255, 0, 0), 2)

face = gray[y:y + h, x:x + w]

face\_resize = cv2.resize(face, (width, height))

cv2.imwrite('% s/% s.png' % (path, count), face\_resize)

count += 1

cv2.imshow('OpenCV', im)

key = cv2.waitKey(10)

if key == 27:

break

After creating the datasets,the camera should start recognizing the person,which is given in the database,

**For face recoginition:**

import cv2, sys, numpy, os

size = 4

haar\_file = 'haarcascade\_frontalface\_default.xml'

datasets = 'datasets'

print('Recognizing Face Please Be in sufficient Lights...')

(images, lables, names, id) = ([], [], {}, 0)

for (subdirs, dirs, files) in os.walk(datasets):

for subdir in dirs:

names[id] = subdir

subjectpath = os.path.join(datasets, subdir)

for filename in os.listdir(subjectpath):

path = subjectpath + '/' + filename

lable = id

images.append(cv2.imread(path, 0))

lables.append(int(lable))

id += 1

(width, height) = (130, 100)

(images, lables) = [numpy.array(lis) for lis in [images, lables]]

model = cv2.face.LBPHFaceRecognizer\_create()

model.train(images, lables)

face\_cascade = cv2.CascadeClassifier(haar\_file)

webcam = cv2.VideoCapture(0)

while True:

(\_, im) = webcam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

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

cv2.rectangle(im, (x, y), (x + w, y + h), (255, 0, 0), 2)

face = gray[y:y + h, x:x + w]

face\_resize = cv2.resize(face, (width, height))

prediction = model.predict(face\_resize)

cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)

if prediction[1]<500:

cv2.putText(im, '% s - %.0f' %

(names[prediction[0]], prediction[1]), (x-10, y-10),

cv2.FONT\_HERSHEY\_PLAIN, 1, (0, 255, 0))

else:

cv2.putText(im, 'not recognized',

(x-10, y-10), cv2.FONT\_HERSHEY\_PLAIN, 1, (0, 255, 0))

cv2.imshow('OpenCV', im)

key = cv2.waitKey(10)

if key == 27:

break

after recognizing the face ,this is taken to next level .now the servo motor should run when the person is recognized and matched with the face in database ,if not it should not run ,this servo motor is fixed to the door ,where the person is allowed if he is recognized and matched with the face in the data base.

import RPi.GPIO as GPIO

import time

import numpy as np

import cv2

haar\_file = 'haarcascade\_frontalface\_default.xml'

datasets = 'datasets'

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(17, GPIO.OUT)

p = GPIO.PWM(17, 50) # GPIO 17 for PWM with 50Hz

p.start(1) # Initialization

rate = 1

inc = 1

cap = cv2.VideoCapture(0)

print("camera is initialized")

count = 0

face\_cascade = cv2.CascadeClassifier(haar\_file)

webcam = cv2.VideoCapture(0)

try:

while True:

(\_, im) = webcam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

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

cv2.rectangle(im, (x, y), (x + w, y + h), (255, 0, 0), 2)

face = gray[y:y + h, x:x + w]

face\_resize = cv2.resize(face, (width, height))

# Try to recognize the face

prediction = model.predict(face\_resize)

cv2.rectangle(im, (x, y), (x + w, y + h), (0, 255, 0), 3)

if prediction[1]<500:

cv2.putText(im, '% s - %.0f' %

(names[prediction[0]], prediction[1]), (x-10, y-10),

cv2.FONT\_HERSHEY\_PLAIN, 1, (0, 255, 0))

else:

cv2.putText(im, 'not recognized',

(x-10, y-10), cv2.FONT\_HERSHEY\_PLAIN, 1, (0, 255, 0))

cv2.imshow('OpenCV', im)

key = cv2.waitKey(10)

if key == 27:

break

if (im==1):

print("you can enter inside")

p.ChangeDutyCycle(12.5) #180

time.sleeep(2)

p.ChangeDutyCycle(7.5)

else:

print("your entry is restricted")

time.sleep(0.2)

ret, frame = cap.read()

#time.sleep(0.2)

cv2.imwrite("/home/pi/Desktop/project/datasets/%d.png" %count, frame)

count+=1

except KeyboardInterrupt:

p.stop()

GPIO.cleanup()

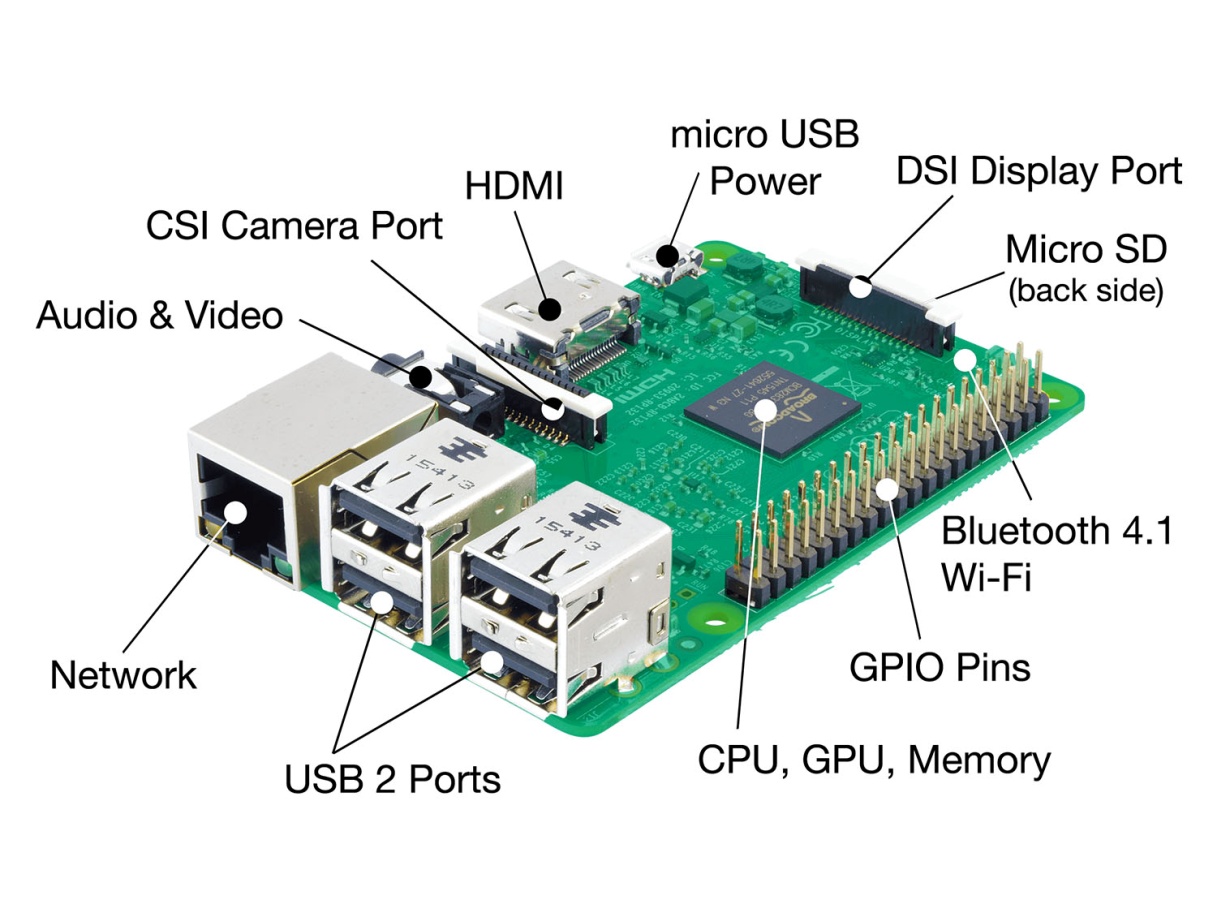
cap.release()

cv2.destroyAllWindows()

**Hardware implementation:**

**RASPBERRY PI3:**

Raspberry Pi 3 is the third generation Raspberry Pi. It is a miniature marvel, packing considerable computing power into a footprint no larger than a credit card. The processor at the heart of the Raspberry Pi system is a Broadcom BCM2837 system-on-chip (SoC) which houses a 1.2 GHz Quad Core ARM Cortex-A53 processor. The vast majority of the system’s components, including its central and graphics processing units along with the audio and communications hardware, are built onto that single component along with 1 GB LPDDR2 memory chip at the centre of the board. It is not just this SoC design that makes the BCM2837 different to the processor found in a typical desktop or laptop, however, it also uses a different instruction set architecture (ISA), known as ARM.

****

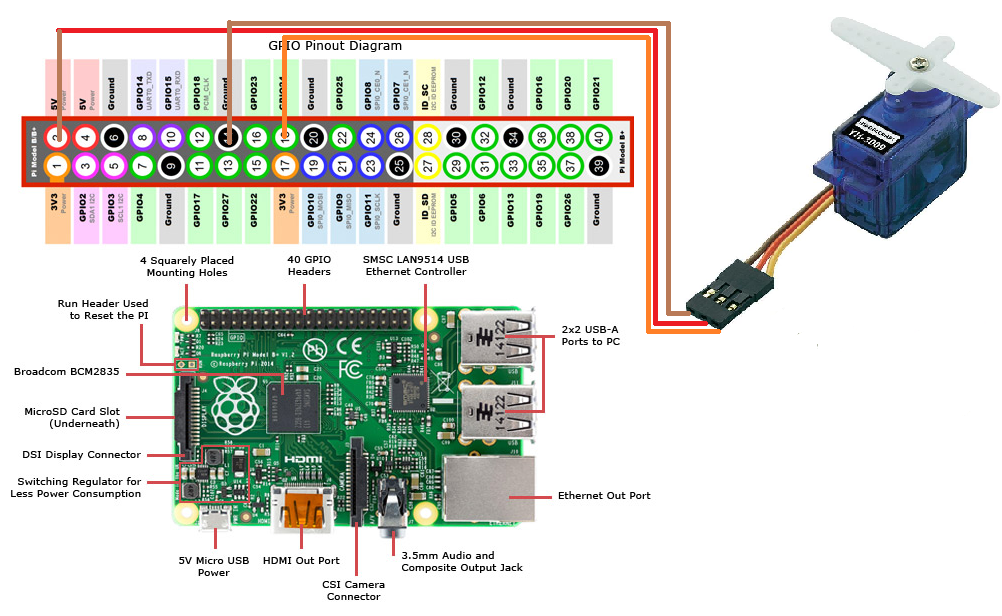
|  |  |  |  |
| --- | --- | --- | --- |
| **I/O index** | **Pi pin** | **I/O index** | **Pi pin** |
| 1 [\*] | 3V3 | 8 | BCM14 |
| 2 | 5V | 9 | GROUND |
| 3 | BCM2 | 10 | BCM15 |
| 4 | 5V | 11 | BCM17 |
| 5 | BCM3 | 12 | BCM18 |
| 6 | GROUND | 13 | BCM27 |
| 7 | BCM4 | 14 | GROUND |

Raspberry pi connections

SERVO MOTOR - PI3

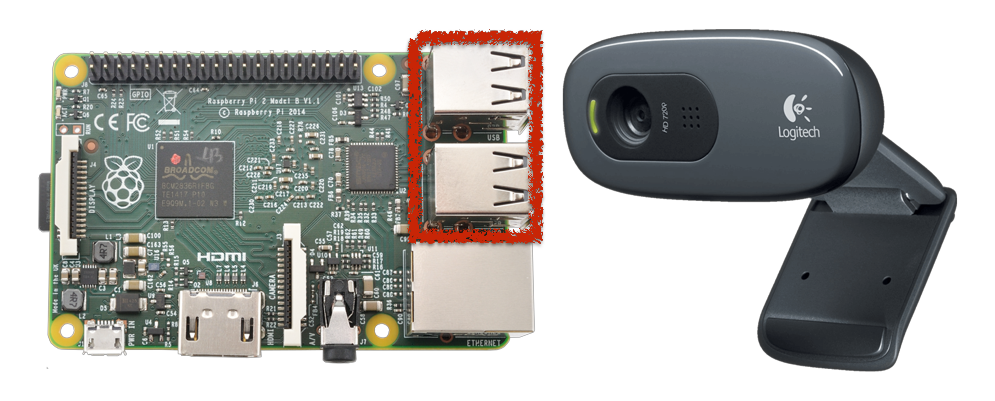
**VCC – 3V3**

**GROUND – 6**

**DATAPIN – BCM17**

LOGITECH CAMERA – PI3

**PLUG – USB PORT**



USB cable is used to give power supply to the Raspberry pi3. Raspberry pi3 has a micro-USB for power.

* The logitech camera is connected to the Raspberrypi to detect the face and to open the door using servo motor.

**ADVANTAGES OF SMART SECURITY SYSTEM :**:

1.Adding Convenience to your Daily Life –

When you convert your home into a smart home, you’ll have all of your products programmed to your specific needs. Additionally being able to control your home, no matter where you are, can be extremely beneficial.

2.Customization –

There are many smart products on the market right now and you certainly don’t need to buy all of them at once. As the consumer it’s up to you to decide which product you want most, determine if you like it, and then add on to your collection of smart home products as you go.

A good product to start with would be a thermostat or home security system if you’re in the market for either of those.

3.Security –

Smart home security systems allow you to view your home no matter where you are. You can have cameras installed, motion detectors, locks, etc, and you will be notified immediately if something is out of the ordinary. Many of these systems will even let you know of any unexpected temperature changes so that you’re alerted if there is a possible fire.

4.Ease of Use –

Almost all smart home products can be installed without much hassle, many of them don’t even require you to bring someone into your home. Additionally if you’re already someone who’s tech savvy, learning how to use most of these products is a breeze.

5.Save Money and the Environment –

Smart homes feature products like thermostats, air conditioners, and lighting. Having the ability to put these things on a timer, or turn them on and off when you’re away from home will likely help you save money on your electricity bills. Many of these products allow you to track your energy usage and expenditures.

**CONCLUSION:**

Overall, this study has been completed properly. All sub-objective of this study has been completed, thus resolved the main objectives and research problems. This system managed to help users to detect human presence around the house precisely. The use of wireless connectivity based on microcontroller facilitates the installation process at home and can prevent anyone hacking into security systems. This system is a system that meets recently smart home applications in order to function in automation situation. During the study, a lot of knowledge and experience have been learned. Besides the exposure on microcontroller, this study also has much exposure to radio frequency technology which is Xbee Pro, where it helps in the transfer process for the wireless signal. This device has helped the development of security systems with its high-tech features.