

FACE RECOGNITION FOR HOME SECURITY SYSTEM BASED ON IOT

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering in ELECTRONICS AND COMMUNICATION
ENGINEERING

by

BOMMANA BOINA TIRUMALA SURYA NARAYANA (37130066)



**DEPARTMENT OF ELECTRONIC AND
COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS
ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND
TECHNOLOGY (DEEMED TO BE
UNIVERSITY)**

**Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

March - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

ccredited with "A" grade by NAAC | 12B Status by UGC | Approved by AICTE
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **BOMMANA BOINA TIRUMALA SURYA NARAYANA (37130066)**, who have done the Project work as a team who carried out the project entitled **FACE RECOGNITION FOR HOME SECURITY SYSTEM BASED ON IOT** under my supervision from December 2020 to March 2021

Internal Guide
Dr. V. VIJAYAKUMAR, Ph.D.

Head of the Department
Dr.T.RAVI, M.E,Ph.D.

Submitted for Viva voice Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, BOMMANA BOINA TIRUMALA SURYA NARAYANA (37130066),

hereby declare that the Project Report entitled **FACE RECOGNITION FOR HOME SECURITY SYSTEM BASED ON IOT** done by me under the guidance of **Dr.V.VIJAYAKUMAR, Ph.D** at Sathyabama institute of science and technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **ELECTRONICS AND COMMUNICATION ENGINEERING**

DATE:

PLACE:

B.T.SURYA NARAYANA

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.N.M. NANDHITHA.**, Dean, School of **ELECTRONICS AND COMMUNICATION ENGINEERING** and **Dr.T.RAVI, M.E,Ph.D.**, Head of the Dept. of **ELECTRONICS AND COMMUNICATION** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.V.VIJAYAKUMAR, Ph.D** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of **ELECTRONICS AND COMMUNICATION ENGINEERING** who were helpful in many ways for the completion of the project.

ABSTRACT

The IOT has made it possible to the most important of feature of any home security system is to detect the people who enter or leave the house. Instead of monitoring that through passwords or pins unique faces can be made use of as they are one's bio-metric trail. These are innate and cannot be modified or stolen easily. The level of security can be raised by using face detection. The proposed face recognition door lock security system has been developed to prevent robbery in highly secure areas like home environment with lesser power consumption and more reliable standalone security device for both intruder detection and for door security. This system is powered by raspberry pi circuit. Raspberry pi electronic board is operated on power supply, wireless internet connectivity by using Universal Serial Bus modem(USB), it includes camera, and a door lock(servo motor). Whenever the person comes in front of the door, it recognizes the face and send to raspberry pi there in raspberry pi the face is registered or not it will check if it get our photo in it, then it unlocks the door, If the face is not registered the door will not open and sent mail to our registered mail that someone is in front of your door.

TABLE OF CONTENTS

ABSTRACT	v
LITERATURE SUREVY	10

CHAPTER NO.	TITLE	PAGE NO.
1. Raspberry pi Basic Hardware Setup		
1.1	Extra Hardware	1
1.2	Connecting Everything Together	2
1.3	Operating System SD Card	3
1.4	Keyboard & Mouse	3
1.5	Display	4
1.6	Power Supply	5
1.7	Cables	6
1.8	Additional Peripherals	
1.8.1.	Internet Connectivity	6
1.8.2.	USB hub	6
1.8.3.	Case	6
1.8.4.	Expansion & Low-Level Peripherals	7
2. Introduction		
2.1	Introduction	7
2.2	Aim of the Project	9
2.3	Project Domain	9
2.4	Scope of the Project	10
2.5	Methodology	10
3. Raspberry Pi Advanced Setup		
3.1	Finding Hardware and Setting up	13
3.2	Serial Connections	13
3.2.1.	First Interaction with the Board	13
3.2.2.	Serials Parameters	13
3.2.3.	Super Easy Way Using GNU Screen	14
3.2.4.	Super Easy Way Using Minicom With Following parameters	14
3.2.5.	GUI Method with GTK Term	14
3.2.6.	Windows User	14
3.2.7.	First Dialog	15

3.3 SD Card Setup	15
3.3.1. Formatting SD Card	15
3.4 Servo motors	16
4. IOT	
4.1 Background	18
4.2 Motivation	19
4.3 IOT Definition	20
4.4 Challenges	23
4.5 IOT Based Middleware	24
4.5.1. Interface Protocols	26
4.5.2. Device Abstraction	28
4.5.3. Central Control, Context Detecting and Management	30
4.5.4. Application Abstraction	31
4.6 Conclusion	32
5. Review of IOT Based Middleware	32
6. Conclusion And Result	55
7. REFERENCES	59

List of Figures

1.2. Connecting Everything Together	2
Block Diagram	3
1.5.1. Raspberry Pi Port Specification	5
3.4. Small R/C Servo Mechanism	18
4.3. Layered Architecture of Internet of Things (IOT)	23
4.5. Functional Components of a Middleware for IOT Based System	26
4.5.2. XML Code to define a Device	30

CHAPTER 1

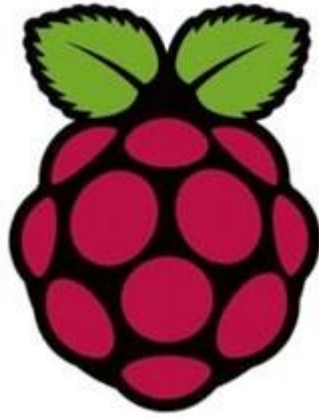
1. Raspberry Pi Basic Hardware Setup

1.1. Extra Hardware

That is something you must do. A processor and graphics chip, RAM, and various interfaces for external devices are all included on the Raspberry Pi board. Some are required, while others are optional. Some are needed. The Raspberry Pi functions similarly to a standard PC and includes a command-input keyboard, a display unit, and a power supply. It also requires 'mass storage,' but the capacity of a regular PC hard drive does not match that of the Raspberry Pi. Instead, we'll use an SD Flash memory card, which is commonly used in digital cameras, and configure it so that the RPi's processor appears to be a hard disk. The RPi can boot from that card in the same way that a PC can boot from its hard disk in Windows (the Operating System loads into RAM).

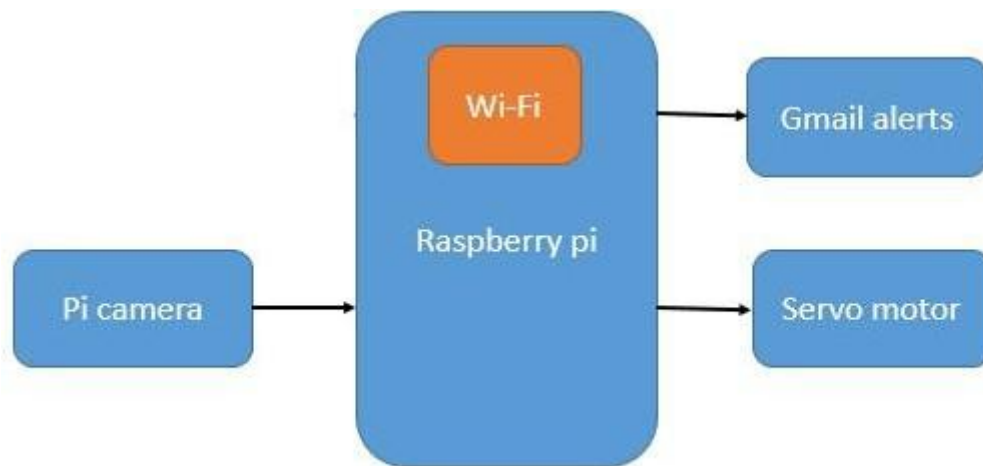
The following are essential to get started:

- SD card containing Linux Operating system
- USB keyboard
- TV or monitor (with HDMI, DVI, Composite or SCART input)
- Power supply (see Section 1.6 below)
- Video cable to suit the TV or monitor used recommended optional extras include:
 - USB mouse
 - Internet connection, Model A or B: USB WiFi adaptor
 - Internet connection, Model B only: LAN (Ethernet) cable
 - Powered USB hub



Case 1.2. Connecting Everything Together

1. Plug the preloaded SD Card into the Raspberry Pi.
2. Plug the USB keyboard and mouse into the Raspberry Pi, perhaps via a USB hub. Connect the Hub to power, if necessary.
3. Plug a video cable into the screen (TV or monitor) and into the RPi.
4. Plug your extras into the Raspberry Pi (USB, WiFi, Ethernet cable, external hard drive etc.). This is where you may really need a USB hub.
5. Ensure that your USB hub (if any) and screen are working.



1.3. Operating System SD Card:

Since the RPi lacks internal storage and a built-in operating system, an SD card containing the Linux Operating System version is needed. You can create your own preloaded card to handle it using any suitable SD card (4GBytes or higher). To avoid arguments about missing images, we suggest that you use a new blank card. SD cards will be preloaded at the RPi Shop.

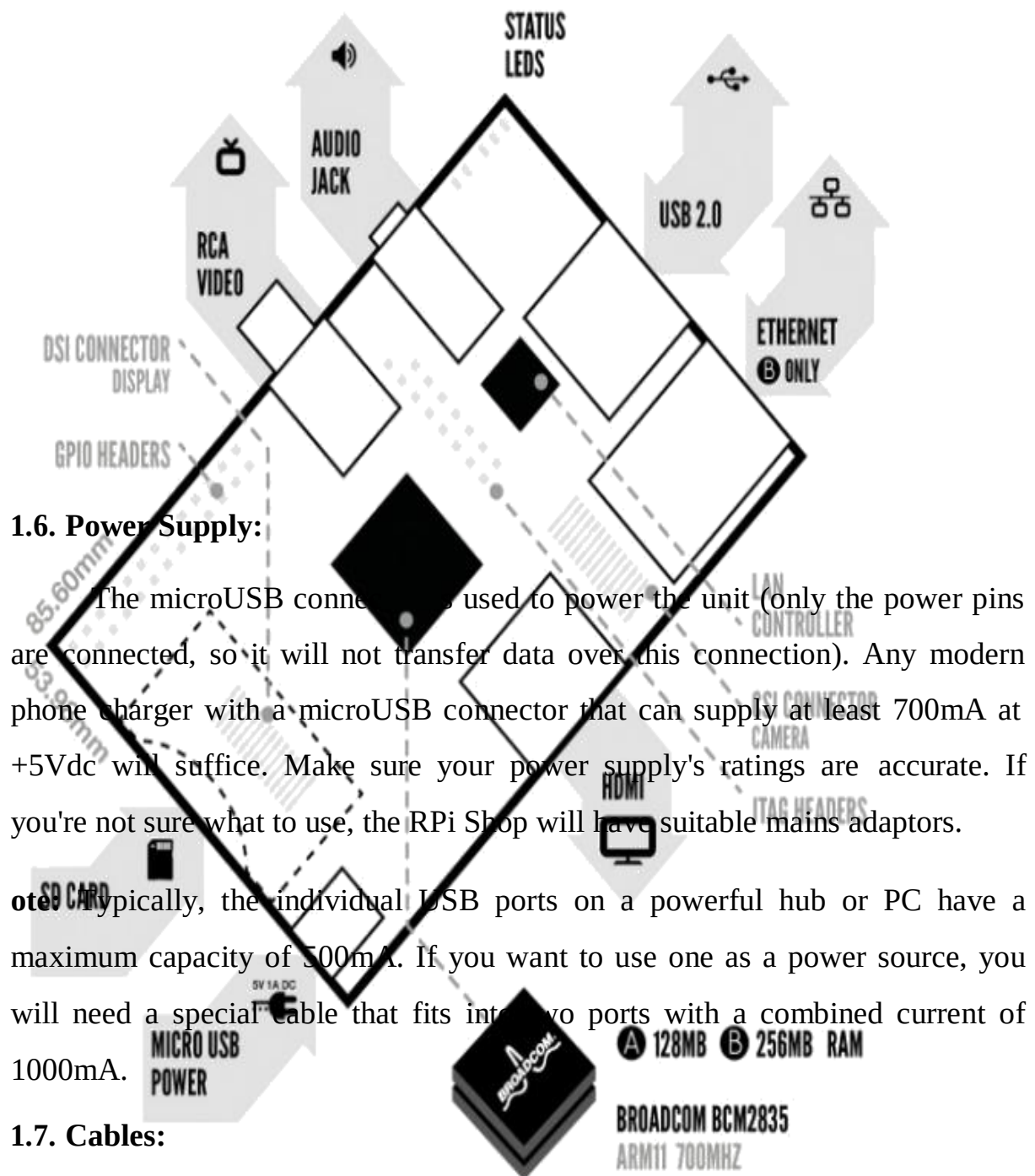
1.4. Keyboard & Mouse:

Most standard USB claws and mice can suit the RPi. Just one USB port for an RF dongle and a working wireless keyboard/mice. To use a Bluetooth keyboard or mouse, you'll need a Bluetooth USB dongle, which uses a single port once again. Model A has one USB port, while Model B has two (typically a keyboard and mouse will use a USB port each).

1.5. Display:

HDMI (High Definition) and Composite are the two main connectivity options for the Raspberry Pi display (Standard Definition). • HD televisions and several LCD monitors can be linked using a full-size 'Adult' HDMI cable and a low-cost adapter. HDMI versions 1.3 and 1.4 support it, and cable version 1.4 is recommended. HDMI is used to output audio and video from the Raspberry Pi, but HDMI is not supported. The yellow to yellow RCA monitor, SCART, and composite video can all be used to link older televisions (using a Composite video to SCART adaptor). Both PAL and NTSC television formats are supported. When using composite video connections, audio from the 3.5mm jack socket can be sent to your TV, headphones, or amplifier. You'll need a sound cable that fits RCA connectors starting at 3,5mm to attach to your TV (red and white). Notice that there is no analog VGA output. This is the requisite relationship, in addition to the new computer monitors. It will not function if you have a monitor with just a 15-pin D-shaped connector.

1.5.1. Raspberry pi Port specification:



You will need one or more cables to connect up your RPi system.

- Video cable alternatives:
 - o HDMI-A cable
 - o HDMI-A cable + DVI adapter
 - o Composite video cable
 - o Composite video cable + SCART adaptor
- Audio cable (not needed if you use the HDMI video connection to a TV)
- Ethernet/LAN cable (Model B only)

1.8. Additional Peripherals:

You can decide to use different other RPi devices including Flash Drives/Portable Hard Drives, Speakers, etc.

1.8.1. Internet Connectivity:

This can be done with an Ethernet/LAN cable or a USB Wi-Fi adaptor. It is possible. The auto sensing of the RPi Model B port means that it is possible to link it to a router or to a device itself (without the need for a crossover cable).

1.8.2. *USB hub:*

You may use a USB hub, which allows the use of several devices to link external devices to the RPi. A powered hub is suggested, which provides the devices with additional power without affecting the RPi itself. It is suggested to have a USB 2.0 model. USB 1.1 is good for mice and keyboards, but might not be fast enough for other accessories.

1.8.3. Case:

Since the RPi is delivered without a case, it is vital for you not to use it where conductive metal or liquids come into contact, unless properly secured.

1.8.4. Expansion & Low-Level Peripherals:

If the low-level interfaces available on the RPi are to be used, make sure you have an appropriate plug for the GPIO header pins. Also, if you consider a specific small-scale project, ensure that the appropriate safety circuits are designed to keep your RPi protected.

INTRODUCTION

2.1 Introduction

Today, human-computer interface technology is increasingly sought after and the machine needs to understand human movements and emotions. When the computer can better understand human emotion, it can improve the performance of the job. It can be a crucial measuring method for conduct research, and robots can be used for developing socially intelligent apps. Emotions constitute intense feelings that are governed by the environment and that play a large part in everyday activities such as decision-making, learning, attention, motivation, handling, perception, planning, comprehension, reasoning, and many more. Text, vocal, verbal and facial expression may be used to recognize the emotions. In 1968, Albert Mehrabian found out that 7% of contact in human-human interaction is supported by verbal signals, 38% by vocal signals and the majority 55% by facial expressions. Analysis of facial expression is also one of the main components for emotional recognition.

2D images are a well-researched area, but they have no real time method that estimates even low-quality images. The majority of work is focused on face pictures. More work is needed on non-frontal images with different lighting conditions, as these global conditions are not standardized in real time.

In our project we propose to use a CMU Multi-PIE database consisting of 2D images with different illuminations and poses that recognizes basic emotions, such as rage, disgust, satisfaction, surprise and neutral. The software system that is de-developed using our proposed method is used on the Raspberry Pi II because it is very compact, light weighted and it needs much less power for robots because the size of the Raspberry Pi II is very small. It can therefore be easily installed on any robot and used for a wide range of applications such as monitoring safety, surveillance of senior citizens, monitoring of vital ICU patients, customer satisfaction and many others.

2.2 Aim of the project

The main purpose of the project is to create a home security facial recognition device to lock or unlock the door by IoT. This helps improve door access safety at home

2.3 Project Domain

The web of articles is also called a remote system between objects. The term IoT shows different advances and research teaches that allow the internet to relate to the present physical reality of the posts. We used the Pi 2 raspberry as a single PC board in the UK created by a pi raspberry facility.

2.4 Scope of the Project

The project was launched and the IoT method was used for effective door access control systems with deep learning methodology. Raspberry Pi is a small computer board programmable and is used as the key control unit in the field of face recognition, youth and locking device. The picture of the individual in front of the door is collected using the camera. IoT allows the user to manage the access of the door.

2.5 Methodology

Collect the image from the camera and then detect the face. We use an image hair algorithm and often use machine training techniques in order to measure the image in particular. Marks of our system are located on the raspbery pi with the dlib library.

LITERATURE REVIEW

Instead of using GSM facilities, the author[1] suggested using Raspberry Pi and relaying in real time for face recognition. Since the background light and ambient light conditions were not regulated, the work was restricted.

The author[2] has proposed a three-subsystem door lock system to be unique facially recognized, face recognition, and automatic door access control. Face recognition is modified using principal component analysis. For the known person in charge of the microcontroller, the door will unlock, and the unknown person will receive an alert. The input photos are taken continuously by the web camera until Demerit of this computer presses the 'stop camera' button. Someone must check the photographs or status of an unauthorized person on the spot and take appropriate action. Since the microcontroller is connected to the Computer, if the PC fails or the feature fails, the whole system will not work.

If someone comes to the door via e-mail or Twitter, the author[3] has implemented a surveillance system that allows the user to see who is standing at the door using a remote camera. Photographs of the person have been taken, and e-mails have been sent. They said that the door could not be operated remotely by the user. They discovered that this system was useful in preventing unauthorized access. This work was restricted in that the generated notice was sent to the postal and twitter accounts, but if the customer did not have an internet connection on their computer, they could not check the e-mail and would not know that any unauteur attempted to access the door.

The author [4] suggested that the embedded image selection method be run on a Raspberry Pi. They registered the image and compared it to the database during this work, but due to the ambient light, the computer was unable to operate properly.

The author [5] suggested using a face recognition system based on a personal-face approach, using a method of facial recognition and a euclidean distance

method, to compare the image per son with the images in the database. It was a process that was fast, effective, and precise.

The author [6] suggested using viola jones for facial detection and main component analysis for image comparisons to introduce a facesetting automated door access system. It is not stable, and it performs poorly. The work's disadvantage was.

CHAPTER 3

3. Raspberry Pi Advanced Setup (Geeks only)

3.1. Finding hardware and setting up:

You will need a preloaded SD card (with HDMI/ DVI/Composite/SCART input), USB keyboard and Power Service (Power Source) (USB charger or a USB port from a powered USB Hub or another computer). Probably a usb mouse, case and a usb hub are also needed (a necessity for Model A). The RPi demand will be reduced by a powered USB hub. You need an Ethernet/LAN cable (Model B) or USB wireless adapter to link to the Internet (either model). It is recommended to attach the power when all is ready for installation.

3.2. Serial connection:

The Serial Port is an easy and straightforward way of connecting to the RPi. The communication is easy to install and is usually usable even before boot time, depending on byte-sensitive data transmission.

3.2.1. First interaction with the board:

Connect the serial cable to the COM port in the RPi, and connect the other end to the COM port or USB Serial Adapter in the computer.

3.2.2. Serial Parameters:

To link to the RPi, the following parameters are needed. All but Port Name and Speed parameters are default and may not be necessary. Port Name: The name of the various types of serial connectors is automatically assigned by Linux.

3.3. SD card setup:

Now, on an SD card, we want to install a GNU/Linux distro and make room for our stuff. The SD or SDHC card can be employed. In the latter case, please, be sure you also help SDHC with your PC card reader. Please realize that the ARM port for the distributor you want to use is not an x86 processor, but an entirely different architecture called ARM.

3.3.1. Formatting the SD card via the mkcard.txt script:

1. Download mkcard.txt .
2. `$ chmod +x mkcard.txt`
3. `$ ./mkcard.txt /dev/sdx`, where x is the letter of the card.

3.4. Servo Motors:

An automatic servo system uses error-sensing negative feedback to correct the performance of a process. Only if systems can track mechanical position or any other parameters with feedback or error correction signals is the term accurate. Feedback on a closed loop, for example, is used in car cruise control, classifying it as a servomechanism.

RC servos are hobbyistic remote-control devices that are used for a variety of mechanical systems, most often on radio-controlled versions. Because of their low cost, reliability, and ease of microprocessor control, RC servos are frequently used in small-scale robotics.

An electric motor mechanically integrates the RC servos into a potentiometer. The PWM signals are sent to the servo by a standard Pulse Width Modulation (PWM) receiver. The electronics within the servo translate the pulse width in a specific location. The engine is pushed to the value of the controlled position when the servo is ordered to rotate.

The RC servos use a 0.1" (female) three-pin spacing jack that corresponds to the standard pin spacing of 0.025." (which should be gold-plated, incidentally). The most popular order is signal, +voltage, field. The normal voltage is 6 VDC, but a few servos have been seen at 4.8V and 12V as well. The control signal is an optical PWM signal with a frame rate of 50Hz. Within every 20ms, an active, high digital pulse controls the spot. The pulse has a nominal duration of 1.0ms to 2.0ms, with 1.5ms always being key. Pulse widths outside of the normal range can be used to move the servo "over flight." This PWM signal is often (incorrectly) referred to as Pulse Position Modulation (PPM).

The servo is controlled by three wires: ground, strength, and energy. The servo shifts in response to the control cord pulses, and the angle of the actuator arm is fixed.

The servo anticipates a pulse every 20 ms in order to obtain accurate angle information. The servo pulse width defines the range of the servo's corner motion.

Typically, the servo is set to 'neutral' or 45° by using a 1.5 ms wide servo pulse, a 1.25 ms pulse at 0°, and a 1.75 ms pulse at 90°. The physical bounds and schedules of servo hardware differ depending on brand and model, but the angular motion of the general servo travels between 90° and 120°, with the neutral position almost always exceeding 1.5 ms. For all analog hobby servos, this is the "standard servo pulse mode."

RC servos are usually supplied by the receiver, which in turn powers the battery batteries or the electronic speed controller (ESC) (BEC). The most popular battery pack is a Ni-Cd, NiMH, or lithium-ion battery (Li-Po). Various voltage ratings are available, but 5 V or 6 V are the most widely used.

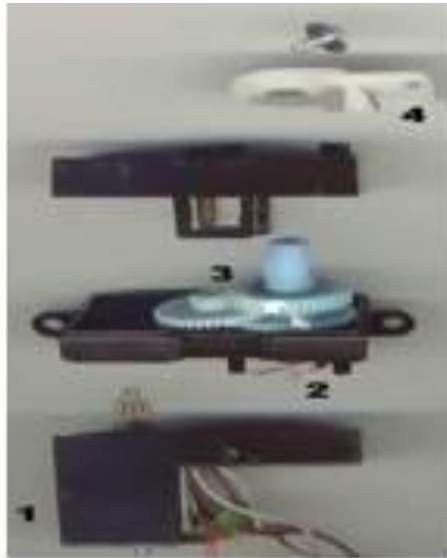


Fig. 3.4 Small R/C servo mechanism

1. Electric motor 3. Reduction

gear

Position feedback 4. Actuator

2. potentiometer arm

CHAPTER 4

IOT

This chapter's structure is as follows: The motives for this work are described in section 1.1. The study goals and questions in Section 1.2 are presented. The steps necessary to achieve our goals are described in Section 1.3. The layout of the report is shown in Section 1.4.

4.1 Background

The aim of the Future Internet is to create an infrastructure that allows people to instantly access the physical world and its objects. Physical objects can be used in a variety of applications, such as e-health, warehouse management, and so on. Each application domain can require different types of physical devices. Each physical device can have its own set of specifications for communication. To achieve the future Internet goal, a layered view that enables data access is needed. The Internet of Things (IoT) is a vision that seeks to integrate a layered system into a world of digital information.

The word 'internet of things' contains two terms: internet and things. The Internet is a global network infrastructure that supports a variety of communication protocols that are versatile and configurable. Things are actual or virtual objects, computers, and knowledge that use intelligent interfaces to provide identities, physical characteristics, and virtual personalities. 1st. An abstract sensor device that contains metadata to identify and locate the corresponding sensor nodes, which can be represented by a virtual entity, for example. As a result, IoT refers to data that can be sent from the real world to the Internet.

Middleware is a software interface that communicates with and manages data between the hardware and application layers[2]. The function of middleware is to present a single programming model for communicating with computers. A middleware hides the issues we face when dealing with heterogeneity and distribution in devices [3].

4.2 Motivation

The IoT-based system is in charge of presenting knowledge about an area to a non-expert user. The IoT framework can be used in a variety of ways, which means that many heterogeneous devices can be discussed. As a result, non-expert users' contact with heterogeneous equipment is an important concern for the implementation of the IoT-based architecture. This issue could be solved by putting a middleware layer between devices and non-expert users. This layer masks the variety of devices from the user's viewpoint and provides end users with clear access to devices.

The definition of the construction of machine abstractions has been addressed in literature. Middleware found in the literature can support device interaction, but it does not support low-level machine configuration [4].

4.3. IoT definition

In this section, we define various Internet of Things terms. There is also an explanation of the layered IoT architecture. The Internet of Things industry has quickly become aware of the importance of communicating with different machine types. IoT would have an impact on business and society by integrating physical devices into information networks[8]. The effects of IoT can be seen in a variety of ways, especially for private and commercial users. From the viewpoint of a private user, IoT is affecting job and personal areas, as well as intelligent homes and offices, e-health and assisted living. The market consumer factor can have an impact on areas such as automation and industrial development, logistics, business process management, and smart transportation of people and products. [9] Formalized paraphrase The following are examples: [9] Formalized paraphrase

IoT integrates physical structures into communication networks. The Internet of Things (IoT) refers to the general infrastructure used to manage this information network, which includes software, hardware, and services. The data on the physical characteristics and information of the integrated physical things will be shared. ID technologies, such as RFID, can be used to uniquely classify each device [10].

"The Internet of Things (IoT) is the global IoT infrastructure for the information society that connects advanced networks, existing and evolving interoperable IT and communications technologies, and (physical and virtual) services."

[11] Formalized paraphrase [11] Formalized paraphrase

According to the Stuff-Architecture 2 Internet (IoT-A), "the notion of a global interconnected continuum of machines, things, and artifacts generally originated with RFID technology, and this concept has been significantly extended to the current approach, with a multitudinous array of heterogeneous objects interacting with the physical world."

[12] Formal paraphrase [12] Formal paraphrase

IoT has a layered architecture that is structured to meet the needs of various industries, businesses, and societies. Fig. 2.1 depicts a generic layered IoT architecture with five layers, which is discussed below: [13] Formal paraphrase
The report is no longer available.

Access Gateway layer

This is a hardware layer that includes built-in systems, RFID tags, sensor networks, and all other types of sensors. This hardware layer can perform a number of functions, such as data collection from a computer or environment, data processing, and communication support.

Gateway Access Layer

This layer is in charge of data handling, as well as the publication and subscription of the topic's resources, message routing, and platform communication.

Middleware layer

This layer performs many critical functions, including data aggregation and filtering collected from hardware devices, data discovery, and user access control for applications.



Figure 4.3. Layered architecture of the Internet of Things (IoT)

4.4 Challenges

For various purposes and regions, IoT-based systems may be used, so that we face different challenges. In this segment, we clarify some of the challenges in research[10]: The following:

Technological advanced

Hardware efforts are needed for the development of embedded equipment, sensors, actuators and (passive and active) recognition technology, as an IoT-based system must be capable of collecting enough real-world information through the use of a wide range of equipment and environments. More work remains to be done to attach and deploy heterogeneous devices in IoT applications and support new ones.

Technologies for networking

Different networks, such as mobile, wLAN and wireless networks, connect stuff in IoT. These networks allow bi-directional communication between real-world objects, applications and services used in IoT applications at different levels. This highly distributed framework can interconnect with low power consumption, while distributed data can lead to problems with privacy.

System of middleware

We have heterogeneous networks and computers in IoT. With new technologies, their heterogeneity may theoretically develop. We should protect their heterogeneity to encourage the usage of these devices with IoT applications. So we must build a stable, scalable and semantically enriched middleware to deal with system heterogeneity.

Products of the platform.

They support an integrated superior management of all participating devices, ensuring scalability, high availability, and a stable and safe performance of the required system features.

We will continue to concentrate on the middleware challenges, so we look for the functionality of the application layer in IoT-base systems that a middleware can provide.

4.5 IoT function Ultra middleware

The middleware systems used on IoT-based systems were examined by Bandyopadhyay S. et. al.[1]. They classify the necessary middleware functionality to handle interaction with different devices in four functional components: (1) interface protocols, (2) device abstraction, (3) central control, context detection, and application abstractions (5) context-detection & management (shown in Fig. 2.2). These components are explained in detail below.

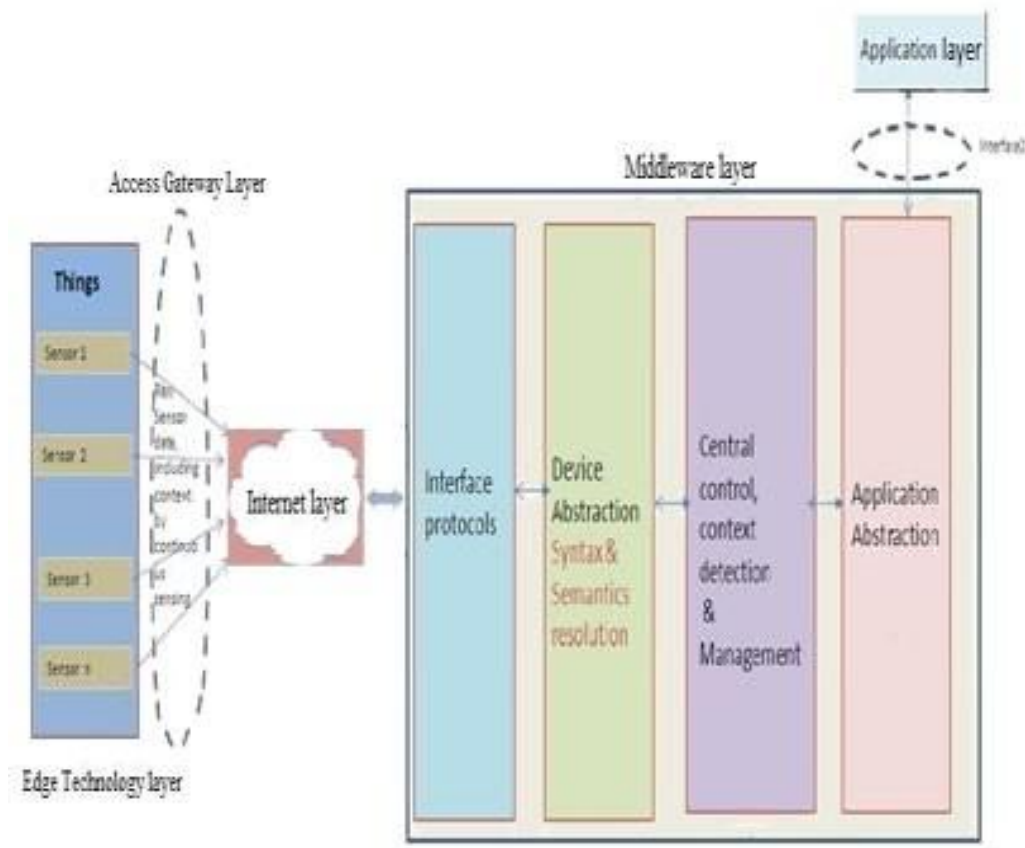


Figure 4.5. IoT-based systems functional components

4.5.1 Protocol interface

The technical interoperability is provided by this part. The ability of two systems to interact with one another using the same protocols implies interoperability in the sense of interface protocols. Technical interoperability is characterized as a combination of modules, systems and platform hardware or software that enables communication between machines and machines in accordance with ETSI (European Telecommunication Standards Institute). This type of interoperability is also focused on the protocols (communication) and infrastructure required for the operation of those protocols[14].

The IP portion defines protocols for information exchange between different networks, which can operate on the basis of different communication protocols, to

```
<dpws:ThisModel>
  <dpws:Manufacturer>ACME Manufacturing</dpws:Manufacturer>
  <dpws:ModelName xml:lang="en-GB" >
    ColourBeam 9
  </dpws:ModelName>
  <dpws:ModelName xml:lang="en-US" >
    ColorBeam 9
  </dpws:ModelName>
</dpws:ThisModel>
</wsx:MetadataSection>
<wsx:MetadataSection
  dialect="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/ThisDevice"
>
  <dpws:ThisDevice>
    <dpws:FriendlyName xml:lang="en-GB" >
      ACME ColourBeam Printer
    </dpws:FriendlyName>
    <dpws:FriendlyName xml:lang="en-US" >
      ACME ColorBeam Printer
    </dpws:FriendlyName>
  </dpws:ThisDevice>
```

To facilitate technological interoperability. This component handles the basic networking of physical and data connection, network, transportation and often the TCP/IP stack application layer.

We can use a wrapper for every device to convert the device-supported protocol to a standard protocol to deal with the heterogeneity of the devices. You may put this wrapper on the side of the computer or the middleware. We should place the wrapper on the middleware side if we want to have a direct contact with devices. Devices typically have limited computer process capacity, which is why middleware wrappers have to be implemented. In comparison, an intermediate wrapper may be established between the middleware and the devices in the event of indirect contact with devices. It is possible to support direct and indirect interactions with the interface Protocol portion.

4.5.2 Device Abstraction (DA)

This component ensures that the application component interacts with the system in an abstract manner. Summarized and semantic interoperability is described as follows in this abstraction: ETSI [14]:

Data formats are synonymous with syntactic interoperability. Communication protocols transmitted messages must have a specified syntax format and encoding format that can be interpreted by high-level transfer syntax, such as HTML and XML.

The significance of the message material that is understandable to human beings is generally related to semantic interoperability. Interoperability at this level therefore means that the significance of content (information) exchanged between people is normal. interoperability at that level means However, as DA does not explicitly communicate with human interoperability, DA has to have this universal understanding for applications in the sense of semantic interoperability.

Semantic interoperability depends on semantic models tending to be unique to the domain. For example, the Service Oriented (SOA)[15] middleware can only be used to provide semantic interoperability with Web Services Devices Profile (DPWS)[16] Any type of computer refers in this context to a distinct type of service Two general functionalities include DA components: (1) requesting devices to execute those features and (2) defining and configuring devices The XML format shown in code 2.1 is used by DPWS.

4.5.3 Central control, Context detection & Management (CCM)

Context describes the situation of an entity which may be the location, individual or object important to, and interacting with, the user[1]. The functional CCM portion supports context-aware computation that takes the context of the system-interacting entities into account. Context-conscious to function in intelligent environments must be a middleware for IoT-based systems [1]. Intelligent environments refer to a physical world that is richly and invisibly linked with sensors, actuators, displays, and computer elements,

seamlessly integrated into everyday objects and connected through a continuous network[17]. Context-awareness requires two functionalities: 1) context identification, consisting of the collection of knowledge and information from tools that can affect the computation.

2) Processing the context, use of the information collected to perform or decide a task.

4.5.4. Application Abstraction

This functional aspect offers an interface to communicate with devices for both high-level applications and end users. This interface can, for example, be a RESTful interface or be used in some query-based languages.

In order to perform data management and data monitoring services, A. Malatras. et. al. provide SOA based middleware[18]. This middleware uses a RESTful interface, so that high-level applications can communicate with the sensors through HTTP: This middleware uses a RESTful interface: (1) DELETE to delete an existing resource, (3) POST to build a new resource and (4) PUT to update the existing resource. For example, by sending a GET request via the URL „http://{hostname}/REST/{version}/DomainTaskResult/id'," a client application gets the result of the domain mission. The reference to a "id" in this URI produces a specific result.

An end-user can use a query language for interacting with devices in the TinyDB middleware [19]. For example, the following query can be sent to obtain the id of a sensor (nodeid) close by and a feel of temperatures within the last ten seconds before the query is executed by this sensor:

Time FROM Sensors Choose Nodeid

Time SAMPLE 1 for 10s SAMPLE

4.6 Conclusion

In this chapter IoT has been described and common IoT layers have been established. In addition, we also addressed the IoT-based systems reference middleware architecture. Bandyopadhyay, S. et. al. suggested this architecture on the basis of a study about the current IoT-based systems middleware frameworks[1].

Chapter 5

Review of IoT-based Middleware

The term IoT refers to a broad range of technologies and research topics, including distributed computing and information administration. Many of the IoT investigations we find in the literature refer to the field of computer technology. World Wide Web Consortium (W3C)³ describes computer comprehension as a vision for our future style of computing. In this perspective, computer systems are incorporated seamlessly into our daily lives, anticipate our needs and provide us with the necessary resources and knowledge in every way. The aim of Pervasive Computing is to facilitate human life by using available computing technologies[20]. This is why scientists are encouraged to create new technologies for daily life.

Since IoT is virtually identical to Pervasive Computing, we have studied IoT and Pervasive systems literature. We had the following approaches to pick the middleware for discussion:

1) Standard IoT middleware such as ASPIRE[22] GSN[23] has been examined. We chose HYDRA [24] as the best popular and most well-documented middleware compared to the above-mentioned. Furthermore, the middleware does not support the functional components discussed in Chapter 3, based on the analysis of Bandyopadhyay, S. et. al, [3], though HYDRA supports them.

2) We looked at typical Pervasive system middleware such as [25], [26], [27]. We selected the following middleware to review:

The middleware is based on the creation and manipulation of data collected from sensors. AURA [28]. We strive to facilitate end user and developer setup and deployment. Since we need to encourage data manipulation in order to achieve this objective, we have reviewed AURA, the only computer program that considered data manipulation in our reviewed paper.

TinyDB [19] focuses on collecting system data. Since we need to collect environmental information from different devices in an IoT-based framework, we examined TinyDB, a common middleware.

WiseMID[29] is the only middleware that is unique to the energy saving application among the tested middleware. We have checked this middleware because it is necessary to save system energy.

We have described the middleware in greater detail in this chapter. In addition, in relation to the functional components described in Chapter 2, we examine the middleware discussed.

Software required:

Python

Open CV

PYTHON:

Python is an interpreted high level language that can be used for general purpose programming. Guido van Rossum created Python's first design philosophy in 1991, emphasizing code readability and a syntax that allows programming companies to express ideas in fewer coding lines [25][26], especially by using wide whitespaces. It provides building blocks for simple small- and large-scale programming. [27] Formalized paraphrase The European Parliament's report

Python includes automated memory control as well as a dynamic type scheme. It supports a wide range of programming paradigms, including object, imperative, functional, and procedural paradigms, and it comes with a large standard library. [28] Formalized paraphrase [28] Formalized paraphrase

Python interpreters are available on a wide range of operating systems. The Python benchmark implementation, CPython, is open source software[29] with a community-based development model, as are almost all of its variants.

CPython is in charge of the Python Software Foundation.

Python was conceived in the late 1980s[30], and in December 1989, it was implemented[31] at the Netherlands Centrum Wiskunde & Informatica

(CWI)[32] as a successor to the ABC language (the self-SETL-inspired one) capable of exceptional handling and interfacing with the Amoeba scheme.

[5] [7] Formalized paraphrase [5] Formal paraphrase Van Rossum is now the primary creator of Python. The title bestowed upon him by the Python community represents his ongoing central role in the evolution of Python: the good-bye dictator for life (BDFL). Van Rossum wrote about Python's origins in 1996: [33] Formalized paraphrase In December 1989, I was looking for a "hobby" programming project that would keep me occupied throughout the week leading up to Christmas. My office was about to close, but I only had a home computer and not much else.

I was thinking about a new scripting language recently, as well as an interpreter: an ABC descendant that would cater to Unix/C hackers. I chose Python as the project's title because I was in an amusing mood (and a great fan of Flying Circus from Monty Python).

— Guido van Rossum

On 16 October 2000, Python 2.0 released and had a number of important new features including a cycle detection waste collector as well as unicode support. The development process with this release became more open and supported by the community. [34] [34] [35]

On 3 December 2008, after a prolonged period of testing, Python 3.0 (also called Python 3000 or Py3k) was announced. It is an essential language revision which is not completely backward compatible. [35] [35] [36] But several of his main features have been re-port to the Python Version Series 2.6.x[36] and 2.7.x and Python 3 releases include the 2to3 tool, which automates Python 2 to Python 3. [37]. [37].

The end-of-life date of Python 2.7(known as EOL, the sunset date) was originally fixed for 2015, during which a significant number of existing codes could not be easily carried to Python 3 until 2020.

[38] The report was subsequently released.

[39] The following [39].

The UTF-8 (Windows, PEP 528 and PEP 529) improvements were made to Python 3.6, the new UTF-8 mode was added to Python 3.7.0b1 (PEP 540). (and overrides POSIX locale).

In January 2017, Google revealed that it will be working on an intercompiler for Python 2.7 to Go to boost workload efficiency.

[40].

Features and philosophy

Python is a language with many paradigms. Object-oriented programming and structured programming are completely embraced, and many of the elements of it support functional programming and aspect-oriented programming. [42] [42] Many other paradigms, including contract design[43][44] and logic programming, are provided by extensions. [45] The report [45]

For the purposes of memory management Python uses dynamics and a combination of reference counting and a cycle detection collector. It also includes a late binding (dynamic naming) resolution, which connects methods and variable name when running the program.

The architecture of Python provides some support for Lisp-traditional functional programming. The function contains filter(), map(), and redu(), list understandings, dictionaries, and sets, and generator expressions. The report [46]. [46] This standard library has two modules (literature and function tools), which incorporate Haskell and Standard ML tools. [47]. [47]

In the document The Python Zen (PEP 20), the central philosophy of the language is summarized, which contains:

Readability counts

Python is structured to be highly extensible rather than having all its functions integrated into its heart. This compact modle makes it especially common as an addition to current implementations of programmable interfaces. Van Rossum

was frustrated with the ABC solution, and his idea of a small core language with a large standard bibliothèque and readily available interpreter was the opposite. [30].

Python's philosophy rejects exuberant syntaxes (like Perl's) in favor of simpler, less cluttered grammar while providing the option of coding methodology. "It is not considered a compliment in the Python community to describe anything as 'smart,'" said Alex Martelli. [49] Formalized paraphrase The document [49] Formalized paraphrase Python's philosophy rejects the Perl approach to language design, which states that "there are more than one way to do it," in favor of "there should be one way – and preferably just one way"[48]

]Python developers strive to avoid premature optimization and reject patches for non-critical parts of CPython that include marginal speed improvements at the cost of clarification.

Formalized paraphrase When speed is critical, a Python programmer can move time-critical features from languages like C to extension modules and use PyPy, a Just in Time compiler. There's also Cython, which converts a Python API script into C and calls the Python interpreter directly from the C-level.

Python's creators have made it a priority to make it enjoyable to use. This is reflected in the language's name, which is a nod to the British comedy group Monty Python[51], as well as in the language's frequently playful approach to teaching materials and reference materials such as spam and ovules (from the well-known Monty Python sketch). [52].

A common neologism in the Python community is pythonic, which can have a variety of program-related meanings. The term "pythonic" refers to code that uses Python idioms, is natural and fluent, adheres to Python's minimalist philosophy, and emphasizes legibility. Unpythonic code, on the other hand, is a rough transcription from another programming language that is difficult to understand or read.

Python users and admirers are also known as pythonists, pythonists, and pythonists, particularly those who are well-known or experienced.

Syntax and semantics

Main article: Python syntax and semantics

Python should be a language that can be read quickly. Its formatting is visually disclosed and also uses keywords in English with punctuation in other languages. Curly brackets are not used to restrict blocks, and semicolons after sentences are optional, in contrast to a number of other languages. The situation is less syntactic than C or Pascal and has less exceptions..[56]

Indentation

Main article: Syntax and semantics for python § Indentation Instead of curly brackets or keywords, Python use white space indentation to restrict blocks. The indentation increases after such statements; the indentation decrease means the end of the current block. [57]. [57]. Often this role is also called the off-side.

Statements and flow of power

The statements of Python include (including others): the assignment statement (token '=', the sign is equal). That is different than conventional imperative programming languages and many other features of the language are illuminated by this fundamental mechanism (including the existence of the variables in Python's version). Attribute at C, e.g. $x = 2$, is a copy of numerical value 2 for 'typed variable name x.' The (right-hand) value is copied to a designated storage place where symbolic address is the (left-hand) variable name. The variable's memory is big enough for the declared kind (possibly very large). In the simplest case of Python assignment, $x = 2$ is used for "(generic) name x receives a reference to a separate object of a (int) numeric kind 2 dynamical assignment." The name is called the object binding. As the position of the name does not include the specified value, a variable is not appropriate to call it. Then named objects of highly differing types can be rebounded at any time, including

strings, procedures, complex objects with data and methods, etc. Successive common value assignments of multiple names, e.g. `x = 2; y = 2; z = 2` lead to the assignment of (maximum) three names and a numbered object, both of which are linked to three names. Since a name is a generic reference holder the association of a fixed data form is irrational. At a particular time, however, a name is bound to an object with a type; thus, dynamic typeing is provided.

The `if` sentence, which executes a code block, together with `elif` (a contraction of `else-if`).

The `for` argument, which travels across an iterable object and captures each element for use by the linked block in a local variable.

The `while` statement that executes a code block as long as it is valid.

The `try` declaration, which permits exceptions to be collected and treated in its annexed code block except for clauses; it guarantees that the cleanup code in a block finally is always run irrespective of how the block exists.

The `class` statement which executes a block of code for object-oriented programming and connects its local namespace to a class.

The `def` declaration that defines a function or process.

The statement that contains a code block (from Python 2.5) inside a context manager, for example (for example to acquire a lock before a code block is executed and then released, or to open a file and then shut it down), allows a resource purchase to start behavior (RAII).

The `pass` declaration, which is a NOP. To construct an empty code block, it is syntactically necessary.

The `assert` declaration of affirming the requirements that should apply during debugging.

The `yield` return statement that returns a generator value. `yield` is an operator in Python 2.5 as well. This type is used for coroutines.

The `import` statement used for importing modules that are able to use their functions or variables in the current program. Importing: `importing <module`

name> or from <module name> importing * or importing numberpy from np or numpy import pi to Pie is four ways.

In Python 3, the print expression has been modified to print().

Python supports the optimization of tail calls or first-class follow-ups, which he says will never happen.

[59] [59]

[60] The Supreme Court of Canada However, the extension of Python generators offers stronger support for a coroutine-like feature in 2.5. [61]. The generators were lazy iterators before 2.5, and the generator was unidirectionally transmitted information. Python 2.5 enables information to be transmitted back into a generator function, and from Python 3.3 several stack levels can be transmitted to the information. [62]

Expressions

Some Python terms are identical to C and Java, although others are not:

Addition, subtraction and multiplication are identical, but the dividing activity varies. In Python there are two kinds of divisions. They are division of the floor and division of the whole. [63] [63] The ** operator for exponentiation has also been introduced by Python.

From Python 3.5, we implemented a new infix user for multiplication using the @ operator for libraries such as NumPy.

[64] The report of the European Commission.

[65] The report of the Council of Europe

== in Python, comparing by value, to Java, compared by value numeric[66] and by reference object.

(The equals() method can be used to compare the value in Java of objects.) Python can be used to compare the identities of an object (comparison by reference). Comparisons, for example a <=b <= c can be chained in Python.

Python uses terms rather than abstract `&&`, `||`, `!` used in Java and C, and `or`, `not`, for his boolean operators.

Python has a kind of term called an understanding of the set. Extended list understandings in Python 2.4 to a more general expression called an expression generator. The report [46].

The use of lambda expressions are anonymous functions, but these are restricted since the body may be only one word.

Conditional expressions are provided by Python as `x if c else y`[68] (different to many other languages as operands of the `c ? X: y` operator).

Python distinguishes tuples from lists. Lists are written as `[1, 2, 3]` and can not be used as dictionary keys and are mutable (dictionary keys must be immutable in Python). Duplicates are written as `(1, 2, 3)`, are unchangeable and can therefore be used as dictionary key if all tuple elements are unchanging. The `+` operator can be used to combine two tuples, so that their contents are not modified directly; instead, a new tuple is created which contains the two supplied tuples. Therefore in view of the variable `t` initially equal to `(1,2,3)`, the initial `t` of `+(4,5)` first evaluates the `t+` value `(4,5)`. The yields of the variable are `(1,2,3,4,5)` which is then returned to the `t` value. For tuples in unambiguous situations, parenthesis are optional. [69] [69]

Python has a sequence unpacking in which several expressions are associated identically to the tuple literal forming, each evaluating something which is assigned to (a variable, a writable property etc) and are placed, overall, on the left side of the equal symbol in an assignment sentence. On the right-wing of an equal sign, the statement expects an iterable object to generate the same number of values as the writable expressions given, and iterates by this to assign the respective left expression to each of the values generated. (Necessary quote)

The operator percentage of Python is `"string format."` In a way like C series, `"spam=%s %d"` (`"blah" 2`) is evaluated by `"spam=blah =2"` This functions analogously to printing cords. The `format()` method of `str` class, e.g. `'spam={0}eggs={1},'` has been complemented by Python 3 and 2.6+. (`"blah"`,

2). Python 3.6 has added f-strings to: `blah = blah; eggs = 2; eggs = {eggs} f'spam = {blah}`. [70] The report was released.

Python has various kinds of string literals:

Strings of single or double quotation marks. Strings As opposed to shells from Unix, languages affected by Perl and PERL operate identically, single quotes and double quotes. The backslash (\) is the escape character of both types of string. The interpolation of strings was available as "formatted literal strings" in Python 3.6. [70] [70]

A sequence of three single and double quotation marks that begin and end. They can span many lines and work in shells, Perl and Ruby like documents here.

Raw string variants, which are denoted with r by literally prefixing the string. Therefore, raw strings can be used where literal backslashes are normal, including standard expressions and Windows paths. Escape sequences are not construed. Compare the following in C#: "@-quoting."

The index for arrays and slicing arrays is `a[key]`, `a[start:stop]` or `a[start:Stop:Step]` in Python's lists. Indexes are null and negative indexes relate to the end. Slices take elements from the start index to the stop index, but not included. The third parameter, phase or stride, can save and reverse components. For instance, a copy of the entire list would be returned by `a[:]`, slice indexes could not be included. A shallow copy is any aspect of a slice.

In comparison to languages such as Common Lisp, Schema or Ruby, a distinction between phrases and statements is rigidly applied in Python. This causes some features to be duplicated. Examples include:

Understandings list vs. loops List

Conditional phrases vs. blocks

The built-in functions `eval()` vs. `exec()` (in Python 2, `exec` is a statement); the former is a statement; the former are statements.

Statements cannot be part of the language, so list and all other understandings or expression of lambda cannot contain statements, all expressions are expressions.

One case in particular is that the assignment statement such as `a = 1` cannot be part of the conditional statement expression. This is advantageous if an assignment operator `==` an equivalent operator is prevented from making a classic C error in conditions: If C code is syntactically correct (but possibly non-intended) but `c = 1` is syntactically valid `{ ... }`, Python creates a syntax mistake.

methods methods

Object methods are functions attached to the class of the object; the `instance.method(argument)` syntax is syntactic sugar for `Class.method` for ordinary methods and functions (`instance, argument`). In contrast to the implicit `self` (or `that`) of some other object-oriented programming languages (e.g. C++, Java, Objective-C, Ruby), python methods provide an explicit automatic parameter of access to instance data. [71]. [71].

Python style uses duck type and has objects, but variable names that are untyped. At compilation time, the type limitations are not checked; instead, operations on the object might fail, meaning the object in question is not of the appropriate type. Although dynamically typed, Python is strong and forbidden, rather than quietly trying to make sense of them, for operations that are not clearly specified (for example, adding a number to a string).

Python enables programmers to define their own types using classes that are used for object-oriented programming most frequently. New class instances are built with the class call (e.g. `Spamclass()` or `Eggsclass()`), and classes are metaclass type instances (they themselves are the instance of themselves), which allow metaprogramming and reflection.

Python had two classes until version 3.0: old and new.

[72]. [72]. The syntax of the two types is the same, which is whether the class object is directly or indirectly inherited from (all new-style classes inherit from object and are instances of type). Both types of classes can be used in versions of Python 2 from Python 2.2. Old classes in Python 3.0 have been deleted.

The long-term strategy supports the incremental typing[73], and from Python 3.5, the language syntax allows static types, but the default CPython execution doesn't search. An experimental static type checker called mypy supports the testing of the compile type. [74] [74]

Summary of Python 3's built-in types

Type	mutable	Description	Syntax	example
bool	immutable	Boolean value		True False
bytearray	mutable	Sequence of bytes	bytearray(b'Some ASCII')	bytearray(b'Some ASCII')
				bytearray([119, 105, 107, 105])
bytes	immutable	Sequence of bytes	b'Some ASCII'	b'Some ASCII'
				bytes([119, 105, 107, 105])
complex	immutable	Complex number with real and imaginary parts		3+2.7j

dict mutable Associative array (or dictionary) of key and value pairs; can contain mixed types (keys and values), keys must be a hashable type

```
{'key1': 1.0, 3: False}
```

ellipsis An ellipsis placeholder to be used as an index in

NumPy arrays ...

float immutable Floating point number, system-defined precision

3.1415927

frozenset immutable Unordered set, contains no duplicates; can

contain mixed types, if hashable frozenset([4.0, 'string', True])

int immutable Integer of unlimited magnitude[75] 42

list mutable List, can contain mixed types [4.0, 'string', True]

set mutable Unordered set, contains no duplicates; can contain

mixed types, if hashable {4.0, 'string', True}

str immutable A character string: sequence of Unicode codepoints

'Wikipedia'

"Wikipedia"

"""Spanning

multiple

lines"""

tuple immutable Can contain mixed types (4.0, 'string', True) But we can append elements using

__add__ .

a = (4.0, 'string', True).__add__('hi',))

Mathematics

While different from other languages, the rounding to negative infinity adds continuity. For example, that means that the $(a + b)/b == a/b + 1$ equation is true all the time. It also means that both positive and negative a values have a true equation $b*(a/b) + \text{percent } b == a$. However, retaining the validity of this equation implies it must be in interval $(b, 0)$ if b is a positive integer while the percent b is as expected at a semi-open interval $[0, b)$. [80]

The rounding of a float to the nearest integer is provided by Python. For tie-breaking, the round (0.5) before 3 is -1.0 , round(-0.5) is -1.0 . [81] The report is not available. Round to even is used by python 3: round(1.5) is 2, round(2.5) is 2. [82].

Python permits boolean expressions in a way that is compatible with general usage in mathematics. The expression $a < b < c$, for example, checks whether a is less than b and b is less than c . This expression is interpreted differently by the languages derived from C: first the expression will evaluate $a < b$, resulting in 0 or 1, and then comparing it with c . [83] [83] [Necessary Page]

Python provides comprehensive arbitrary accuracy support. Integers that belong to the python type int, can be transmitted to an arbitrary accuracy, belonging to the python type long where appropriate, from the maximum machine supporting precision (usually 32 or 64 bit). The above have a suffix of "L" in their wording. [84] The report [84] (In Python 3, the difference between int and long forms has been removed; the int class now holds this behavior entirely.) The decimal type/class in the decimal module (from version 2.4) offers arbitrary precision decimal floating point numbers and a range of ring modes. [85] The module fraction form (from version 2.6) provides rational numbers with arbitrary accuracy. [86] [89]

Due to the extensive library of Python mathematics and NumPy a third-party library that enhances native capabilities, it is also used to support issues such as digital data processing and manipulation as a scientific script language.

Libraries

The broad standard library of Python, also referred to as one of its greatest strengths[87], offers resources suitable for a variety of tasks. Many of the mainstream formats and protocols like MIME and HTTP are supported for Internet-facing applications. It includes modules for the development of graphic user interfaces, connections to relational databases, pseudorandom numbers generation, arithmetic with arbitrary decimal precision[88], regular expressions management and unit testing.

Many sections of the standard library are protected by requirements (for example, the WSGI implementation follows PEP 333[89]), but most of the modules do not follow PEP 333[89]. The code, internal documentation and test suites are listed (if supplied). However, since most of the standard libraries are cross-platform Python code, only a couple of modules need to be changed or rewritten.

As of September 2017, there are over 117,000[90] packages with various functions, including the Python Package Index, the official third-party Python software repository.

Graphical user interfaces

Web frameworks

Multimedia

Databases

Development environments

For a comparison of integrated developments § Python see also:

Most implementations of Python (including CPython) have a read-eval-print loop (REPL), so that they may function as a command line translator for which the users sequentially enter the statements and obtain immediate results

Other shells, IDLE and IPython included, provide additional features such as self completion, preservation of session status and highlighting the syntax.

There are web browser IDEs, SageMathe (aimed to improve science and Math Python programs); Python Anywhere, a browser IDE and hosting environment, and Canopy IDE, a Python IDE commercial to highlight Science Calculation, as well as standard Desktop integrated development environments (see Wikipedia's "Python IDE" article).

Implementations

See also: Python Applications List § Python Delivery Implementation of the reference

CPython is Python's reference execution. It is written in C, with some features C99. It complies with C89. [93]It compiles Python programs into a mid-byte[94], and then performs them on your virtual machine. [95] CPython is distributed in a mixture of C and Native Python with a wide standard library. It is available on many platforms, including Windows and state-of-the-art Unix systems. One of the early goals was platform portability

Additional implementation

PyPy's Python interpreter 2.7 and 3.5 is fast, compatible[97]. Its just-in-time compiler improves CPython significantly. [98] [98] A version is being developed to take advantage of multicore processors by using machine transactional memory. [99] The following: [99]

Unstoppable Python, which implements microthreads, does not use the C memory stack, which enables huge concurrent programs, is an important fork for CPython. There's also a PyPy edition stackless. [100].

MicroPython is a microcontroller optimized Python 3 version.

Implementation without help

There have been other just-in-time Python compilers but are not supported now: The goal of Google was to quadruply increase the Python interpreter by using LLVM in 2009 with the objective of increasing its multi-threaded capabilities to thousands of cores. The project was called Unladen Swallow.

[101] The report was submitted.

Psyco is a specialized just-in-time compiler that combines CPython and transforms byte code into machine code during operation. The issued code is specialized in some types of data and faster than Python code.

In 2005 Nokia released a Python translator for the PyS60 series. It consists of several CPython modules and a number of additional modules that can be integrated with the Symbian operating system. To be run on all variants of the S60 platform, the project was kept up to date, with several third-party modules available. The Nokia N900 also offers GTK widget libraries for Python, so that programs can be written and run on the target computer. [102].

Cross-compilers to other languages

There are several compilers to high-level object languages, with either unrestricted Python, a restricted subset of Python, or a language similar to Python as the source language:

Jython compiles into Java byte code, which can then be executed by every Java virtual machine implementation. This also enables the use of Java class library functions from the Python program.

IronPython follows a similar approach in order to run Python programs on the .NET Common Language Runtime.

The RPython language can be compiled to C, Java bytecode, or Common Intermediate Language, and is used to build the PyPy interpreter of Python.

Pyjs compiles Python to JavaScript.

Cython compiles Python to C and C++.

Pythran compiles Python to C++.

Somewhat dated Pyrex (latest release in 2010) and Shed Skin (latest release in 2013) compile to C and C++ respectively.

Google's Grumpy compiles Python to Go.

Nuitka compiles Python into C++ [103]

Performance

A performance comparison of various Python implementations on a non-numerical (combinatorial) workload was presented at EuroSciPy '13.[104]

Development

The development of Python is largely done by means of the Python Enhancement Proposal (PEP) process, the primary means by which significant new characteristics are proposed, community inputs are collected and decisions on design of Python are recorded. [105] [105] The Python group and Guido Van Rossum, the Benevolent Dictator for Life in Python, are examining and commenting on excellent PEPs. [105]

The growth of the CPython reference implementation corresponds to the improvement of language. The python-dev mailing list is the principal platform for the growth of the language. The Roundup bug tracker at python.org discusses specific problems. [106] Development of a self-hosted source code repositories running Mercurial originally took place in January 2017, before Python migrated to GitHub. [107]

Public releases of CPython are available in three different forms, which differentiates between the increase of the version number:

Backward versions, of which code should be broken and manually ported. The version number for the first component is increased. These launches occur rarely — eight years after 2.0 for example version 3.0.

Major or "function" updates are widely compatible, but add new features, approximately every 18 months. The version number of the second component is increased. For several years after its release, each major version has been supported by bugfixes. [108]

Approximately every three months a bugfix patch, which does not include new functionality, is released if enough bugs are resolved in advance from the last release. These releases also fix security vulnerabilities. The version number is increased in the third and last section. [109]

Many alpha, beta and release applicants are also available before and for tests. Although every release has a harsh timetable, it is always postponed if the code is not ready. During the development process and through the ongoing BuildBot Integration Framework, Python's development team tracks the status of the code. [110].

More than 86,000[111] software modules have been added by the group of Python developers to the Python Package Index (PyPI), the official repository of Python-based third-party libraries (by 20 August 2016).

Python is the leading academic conference. Special Python mentoring programs such as Pyladies are also available.

Naming

The name of Python comes from the British Monty Python comedy party, whose language Python founder Guido van Rossum enjoyed. In Python, for example, the metasyntactic variables often used in Python literature are spam and eggs instead of standard foo and bar; for example, the metasyntactic variables are spam and egg instead. [113]There are numerous references to Monty Python routines in the official Python documentation. (114)

The Py- prefix shows that Python is associated with something. Examples of how Python applications or libraries use this prefix include: Pygame, SDL-binding to Python (usually for creating games), Python for S60, Symbian Operating System Implementation; PyQt and PyGTK, both of which bind Qt and GTK to Python; and PyPy, Python Implementation that originally was written for Python Uses.

Principal article: program list Python

Since 2003, Python has always ranked in the TIOBE Programming Community Index in the top ten most popular programming languages. It is the 4th best-known language since January 2018. [116] The report was published in French. In 2007 and 2010, it was selected Program Language of the Year. [117] It is the third most common language with a non-C dominated grammar syntax.

An empirical study found that scripting languages such as Python, for programming problems related to string manipulation and searches in a dictionary, are more efficient than traditional languages such as C and Java, and found that the consumption of the memory was "such as better than Java or C or C++.[118]

Wikipedia, Google,[119] Yahoo!,[121] CERN,[121] NASA,[122] some smaller organizations such as ILM,[123] and ITA are large organizations which use Python.

[124] The report was released. Reddit's social news website is completely developed in Python.

For web applications, Python may use mod wsgi, for example, for the Apache web server as a script language.

[125] A standard API has been created to promote these applications with the Web Server Gateway Interface. The development and maintenance of complex applications are supported by Web frameworks such as Django, Pylons, Pyramid, TurboGears, web2py and Tornado. The client-side of Ajax-based applications is developed with Pyjs and IronPython. As data mapper for a relationship database SQLAlchemy may be used. Twisted is a format for computer program correspondence, which is used by Dropbox, for example.

Library libraries including NumPy, SciPy and Matplotlib make it useful in scientific computing for Python, [110][127] providing domain-specific features for specialized libraries such as Biopython and Astropy. The software SageMath in Python can be used for mathematical purposes, such as algebra, combinatorics, numerical mathematics, number theory and calculus. For numerical and statistical calculation with 2D/3D display by the DMelt project, the PYTHON language reimplemented in Java framework is used. [128][129]

Python has been successfully integrated as a scripting language into many software products, including in the form of end-to-end element process, such as Abaqus, 3D parametric modeller such as FreeCAD, 3D animation packages, including 3ds Max, 3-D Blender, 3ds Max, Cinema 4D, Lightwave, Houdini, Maya. As a pretty printer GNU Debugger uses Python to display complex

structures like C++ containers. Esri encourages Python as the best way to write ArcGIS scripts. [132] It was also used in a number of video games[133] and was first used by the Google App Engine in three programming languages; two were Java or Go. [135] [135] In algorithm trade and in quantitative finances, Python is also used. [136] In online brokering APIs that run in other languages, Python can also be implemented with wrappers. [137]

In artificial intelligence projects Python has been used.

[138] Python is also used for natural-language processing as a scripting language with modular architecture, plain syntax, and rich text processing tools. [142]

Python is a standard component for many operating systems. It can be used from the command line and is equipped with most Linux distributions, AmigaOS 4, FretBSD and NetBSD (terminal). Many Linux distributions use an installer in Python: the Ubiquity installer is used by Ubuntu, and the Anaconda installer is used by Red Hat Linux and Fedora. In its Paket Management Framework, Portage, Gentoo Linux uses Python.

Python is widely used, including for exploit creation, in the information security industry.[143]

[144] Python is the main part of the One Laptop per Child XO Sugar program, which has already been developed at Sugar Labs.[145]

The single board Raspberry Pi computer project took Python as its main LibreOffice user programming language and plans to replace Java with Python. Python can also be used. The Script Provider of Python is a central function [146] as of 7 February 2013, after version 4.0.

Python-affected languages

Many other languages have been inspired by Python's architecture and philosophy:

Boo uses a similar syntax, indentation and a similar model object.[147] Cobra uses a similar syntax and indentation and its Recognition document lists Python first among its influential languages.

[148] Cobra, however, explicitly supports contract design, unit testing and static optional typing. [149]

The Python-inspired syntax of CoffeeScript is a language which cross-compiles to JavaScript.

ECMAScript has borrowed from Python iterators, generators and list understandings.[150]

Go is characterized by "work in a dynamic language like Python's development speed." [151]

The ability to bring the theory of design to Java led Groovy to.[152]

Julia was developed 'by true macro [.. and to be] accessible as Python [and] as fast as C for general programming.'

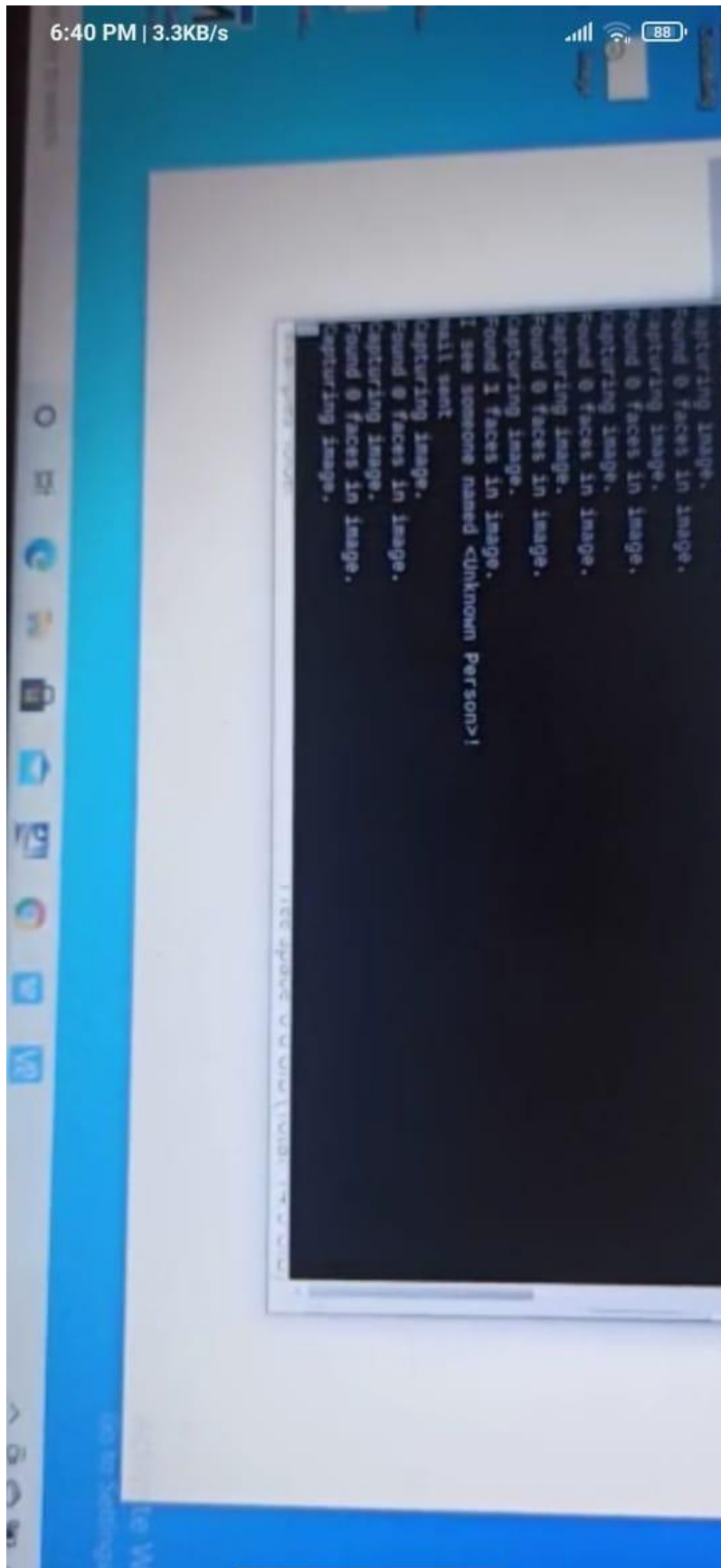
It is possible to call to or from Julia; to make callings from Python from Python, with PyCall.jl and with a Python pyjulia package.

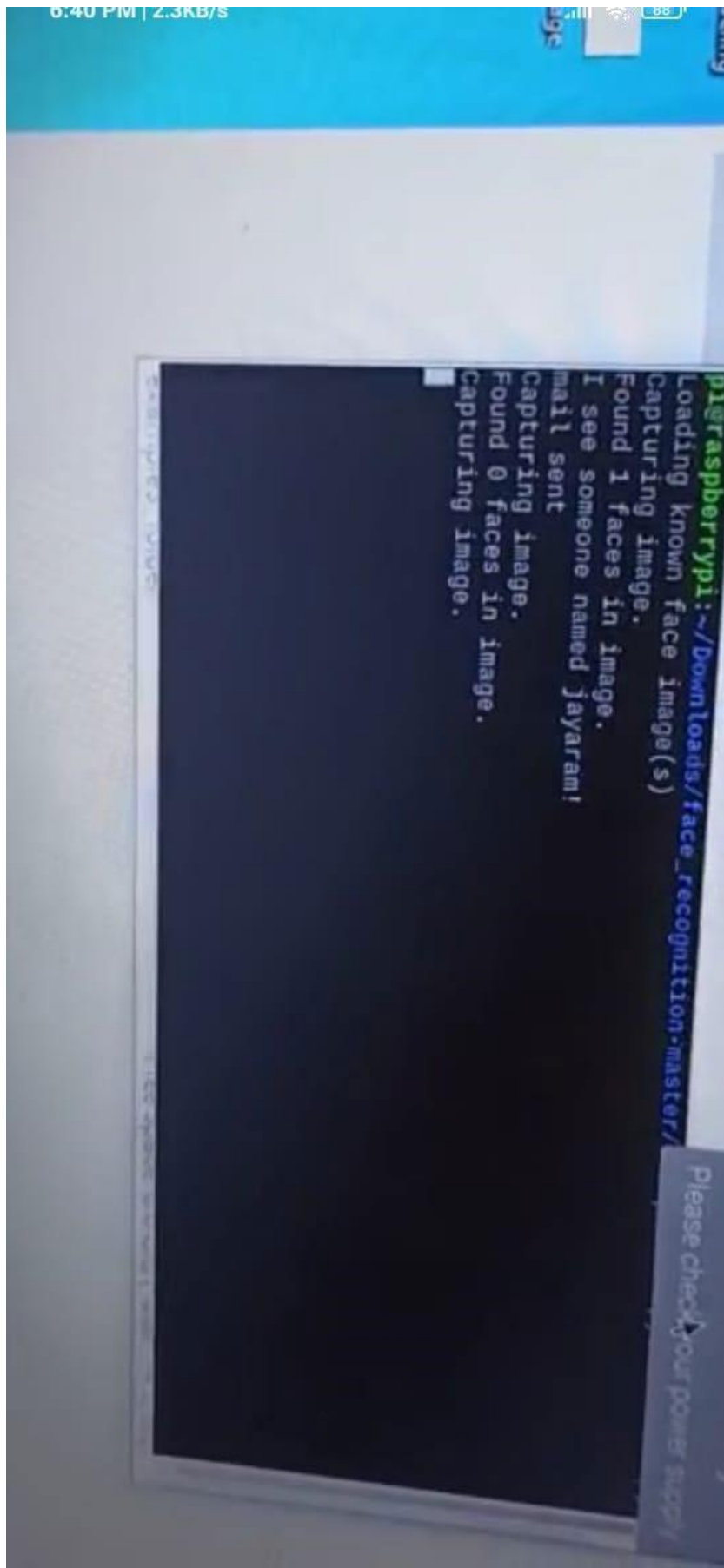
The optional OCaml syntax called the twt is Python and Haskell inspired.[153]

Yukihiro Matsumoto, developer of Ruby, said: "I wanted a more efficient scripting language than Perl, more object-oriented than Python. That's why I choose my own language." [154]

Swift, an Apple-developed programming language, has a certain Python syntax.[155]

Result:







unknown person Inbox



btsuryanarayana007@gmail.com 20 Feb
unauthorised person in the door



btsuryanarayana007@gmail.com 20 Feb
unauthorised person in the door



btsuryanarayana007@gmail.com 20 Feb
unauthorised person in the door



btsuryanarayana007@gmail.com 20 Feb
unauthorised person in the door

↩ Reply

↩↩ Reply all

➦ Forward

REFERENCES:

- [1] Jie-Ci Yang et. all An Intelligent Automated Door Control System Based on a Smart Camera.
- [2] S. Nazeem Basha et all An Intelligent Door system using Raspberry pi and Amazon Web services IOT.
- [3] Richard Grimmer, Raspberry Pi Robotic Projects. Packt Publishing.
- [4] W. F. Abaya, J. Basa, M. Sy, A. C. Abad and E. P. Dadios, "Low cost smart security camera with night vision capability using Raspberry Pi and OpenCV," 2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Palawan, 2014, pp. 1-6
- [5] Priya Pasumarti¹, P. Purna Sekhar "Classroom Attendance Using Face Detection and Raspberry-Pi" International Research journal of Engineering and Technology (IRJET), Volume: 05 Issue: 03. p3-p5,Mar-2018
- [6] S Rajkumar, J Prakash, "Automated attendance using Raspberry pi", International Journal Of Pharmacy & Technology (IJPT), Vol. 8, No. 3, pp. 16214-16221, September 2016.
- [7] T. M. Inc., "Train a Cascade Object Detector," [Online]. Available: <http://www.mathworks.se/help/vision/ug/traina-cascadeobject-d>