A Project Report

On

ARTIFICIAL INTELLIGENCE OPERATING SYSTEM INARMJHA

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE
MINOR PROJECT

By
K.N.S.N GURU CHARAN(1406102)
N.SAI KRISHNA(1406036)
M.KISHORE KUMAR(1406015)
B.SANDEEP REDDY(1406001)

UNDER THE SUPERVISION OF

Dr M.T.U HAIDER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

MAY 2017

CERTIFICATE

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY PATNA

This is to certify that **K.N.S.N Guru Charan**(1406102),**N.Sai Krishna**(1406036),**M.Kishore Kumar**(1406015),**B.Sandeep Reddy**(1406001) have carried out the minor project entitled "**Artificial Intelligence Operating System INARMJHA**" under the supervision of **Dr M.T.U HAIDER**,**Department of Computer Science and Engineering**.This project is bonafide work done by them for the fulfillment of the requirements for the minor project .

Dr M.T.U Haider	Dr Prabhat Kumar
Supervisor	Head of Department CSE

DECLARATION

We hereby declare that this project entitled "**Artificial Intelligence Operating System INARMJHA**" has been carried out by us in the department of Computer Science and Engineering of National Institute of Technology Patna under the guidance of Dr M.T.U. Haider , Department of Computer Science and Engineering , NIT Patna. No part of this work has been submitted to any other institute.

K.N.S.N Guru Charan 1406102
N.Sai Krishna 1406036
M.Kishore Kumar 1406015
B.Sandeep Reddy 1406001

Place: Patna
Date: May 2017

To our professor Dr.M.T.U Haider

Table of Contents

List of Figures

Acknowledgements

We express our deepest gratitude towards our project guide Dr. M.T.U Haider (Department of Computer Science Engineering,NIT Patna) for his valuable suggestions,insightful criticisms and directions for the proposal development of this project.He has always encouraged us to get out of the shell and do something innovative despite our limitations

We are also thankful to all the non-teaching staff other facilities of the Department of Computer Science Engineering, NIT Patna for their support.We also like to thank Shaik Dawood who is a great innovator and dreamer of technology for helping us in the mechanical Design of the automatic door locker system.
Finally, we thank our friends and family members for providing constant encouragement, support and valuable suggestions during the development of the project

1.K.N.S.N GURU CHARAN 1406102
2.N.SAI KRISHNA 1406036
3.M.KISHORE KUMAR 1406015
4.B.SANDEEP REDDY 1406001

Synopsis

Artificial Intelligence are a group of programs that learns a lot without being explicitly coded by hand.These programs are changing the world and helping it to make world a better place to live. One of the most prominent gifts from the Computer Science Engineering to the world are a group of programs that are capable of managing resources known as "Operating System". Operating System are evolving from normal programs to smart Operating Systems that are capable of running organisations. Our project deals with a new branch of AI and Operating system known as "Artificial Intelligence Operating System"(AI–OS).

The aim of this work is to build an AIOS that is capable of running a Smart Home. "Smart OS that Runs a Smart Home".This idea is a huge pool of tasks. So,We had identified a small basic set of tasks that are essential to solve and proposed an interesting approach to solve with the help of Machine Learning algorithms and Operating System Concepts. A Smart Operating System should be able to control devices,should be able to communicate with the user for this there should be external agent between the operating system and user. Now a days most used electronic devices are mobiles. So we build an user friendly Android application which provides as an interface between the user and the operating System. The user can send commands,communicate with the os and also it displays the messages and notifications from the operating System to the user. One of the most important Feature of an Intelligent is to communicate with the user and for this first it should be able to understand the language used by the users. Thus ,here comes the importance of the most popular branch of the AI Natural Language Processing.We have implemented

an Information Retrieval Model. We in this project has answered the interesting questions of how to implement such a model? What is the data set is built one?thus itself bringing a lot of excitement to this project. Once we have created the intelligent programs now the next question arises is "How to incorporate this programs into a OS?". This question is answered with huge amount of research using a deeply and using concepts of implementation view of the operating System

One of the interesting objective that we already stated is control of devices we have answered this question with giving a detailed implementation of intelligent door locking system which provides security by detecting faces that comes around the door and sending a notification to the app the image of the face detected.How can one such implement a system? we had answered carefully this problem with a in depth concepts form Computer Vision,Arduino and Mechanical Gearing Systems. This Project itself is a huge excitement.

Chapter 1

Introduction

What was once just a figment of the imagination of some our most famous science fiction writers, artificial intelligence (AI) is taking root in our everyday lives. We are still a few years away from having robots at our beck and call , but AI has already had a profound impact in more subtle ways. Weather forecasts, email spam filtering, Google's search predictions, and voice recognition, such as Apple's Siri, are all examples. What these technologies have in common are machine-learning algorithms that enable them to react and respond in real time. There will be growing pains as AI technology evolves, but the positive effect it will have on society in terms of efficiency is immeasurable. There is so much potential for AI development that it is getting harder to imagine a future, without it. The main idea behind our project came from Mark Zuckerberg's AI assistant-JARVIS

. Operating system manages all the programs which run on a computer. It allocates memory, takes care of which program to be executed and scheduling of programs, I/0 Management. It is the main program which runs in a computer which controls all other programs. Mainly OS manages every resource in a computer efficiently. Whereas, Artificial intelligence is another set of programs where you actually

code less (you won't program it for each decision it makes rather the program itself figures out what's best and does on its own). Example for AI program, if you have to classify a dog in an image it will be very difficult to code, to recognize each and every dog. But using AI if you give program, some dogs images and it has some minimal instructions which will allow the program to learn the concept and predict. Today in modern day life, the life of humans is so complicated that they need some assistance. While the activities of human are growing up, there is a growing focus to involve technology to save his time. Keeping this in mind, we targeted the problem of managing and controlling a smart home. Smart OS helps the user to control smart home and provide security to it. We designed an Artificial Intelligence Operating System (AIOS) which uses detect faces, controls devices, communicates with user through app.

1.1 JHA

As a solution to the above described problem we created an Artificial Intelligence Character "Jha—An Intelligent AIOS". JHA controls your smart home and it can do the following things for you:

1.Provides security by using a door camera and face recognition technology.

2.Controls all the devices of your home through the local area network .

3.Can talk to you through the app and understand your command in natural language.

4.JHA IS POWERED BY MACHINE LEARNING.

These are the various modules that helps JHA to give its capabilities:

1.Operating system module

2.Face recognition module

3.Communication module

4.Natural language module

5.Devices module

1.Operating system module:

The operating system is built with the help of several open source distributions. A new desktop environment is added. The operating system provides background services for the modules to run.

2.Face recognition module:

The smart camera in front of the home senses (detects) any human face and takes snap of the face and sends it to the app in the users mobile.

3.Communication Module:

The communication module establishes a stabilized connection between user, OS and OS, Arduino (go to devices module).

4.Natural Language module:

This module enables the app to understand users command(voice), understand and reply accordingly.

5.Device Module:

In this module it tells about how connection is established between the smart devices in the home, operating system and Arduino.

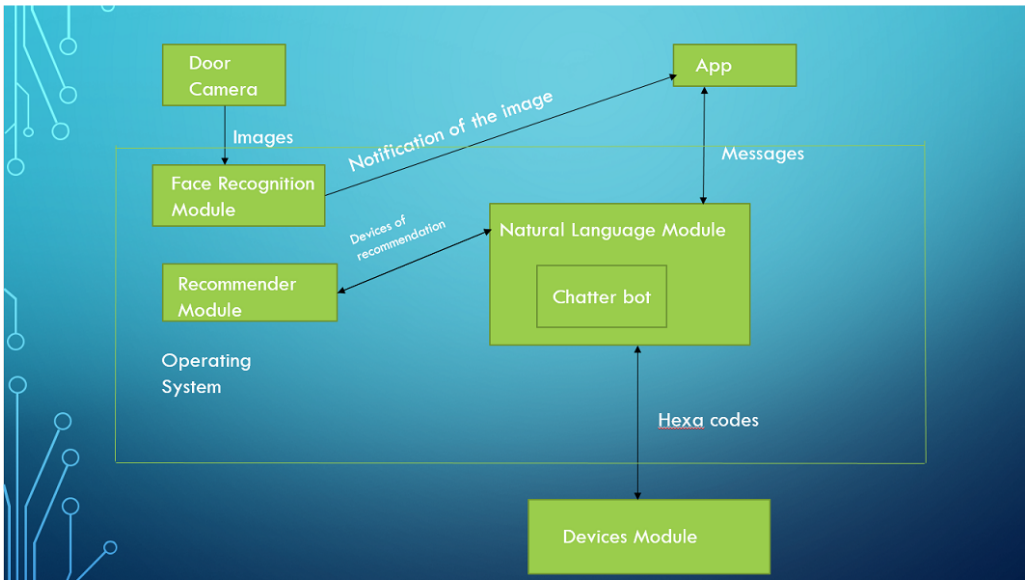

Figure 1.1: Flow Diagram of Project

1.2 Arduino

Arduino is a microcontroller kit for building digital devices and interactive objects that can sense and control objects in the physical world. Arduino also has a rich technology support for executing codes written in various programming languages like C,C++,Python. What can you do with Arduino?? We can program devices to work without the intervention of human. The devices are programmed through arduino. We simulated the working model of door system to provide security to smart home with Arduino and OS.

Chapter 2

Face Recognition Module

In this chapter we will provide a solution to one of our sub problem Face Recognition. How we approach to the solution of this sub problem and what are the problems we face in the recognition and finally we conclude with the question whether Face Recognition works or not?

2.1 Introduction

A Face recognition system is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by computing selected facial features from the image and a face database. It is typically used in security systems and can be compared to other biometrics such as fingerprint. Recently, it has also become popular as a commercial identification and marketing tool.

2.2 FACE RECOGNITION SYSTEM

Facial recognition is a subfield in a larger field of pattern recognition research and technology. Pattern recognition technology uses statistical techniques to detect and extract patterns from data in order to match it with patterns stored in a database. The data upon which the recognition system works (such as a photo of a face) is no more than a set of discernable pixel-level patterns for the system, that is, the pattern recognition system does not perceive meaningful faces as a human would understand them. Nevertheless, it is very important for these systems to be able to locate or detect a face in a field of vision so that it is only the image pattern of the face (and not the background noise) that is processed and analyzed. The first step in the facial recognition process is the capturing of a face image, also known as the probe image. This would normally be done using a still or video camera. In principle, the capturing of the face image can be done with or without the knowledge (or cooperation) of the subject. This is indeed one of the most attractive features of Face Recognition. As such, it could, in principle, be incorporated into existing good quality passive CCTV systems. Steps in Face Recognition System-: 1) Face detection from a image
2) Face identification by comparing the features in the image database

2.3 Face Detection

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. We will try to solve this sub problem using a famous Viola-Jones Algorithm[1]. Viola-Jones algorithm[1] uses HAAR Feature

based Detection

2.3.1 Haar feature-based cascade classifiers

It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect faces in other images. The algorithm is as follows:-

The algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier.

Then we need to extract features from it.

For this, haar features shown in below image are used.

Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle

Now all possible sizes and locations of each kernel is used to calculate plenty of

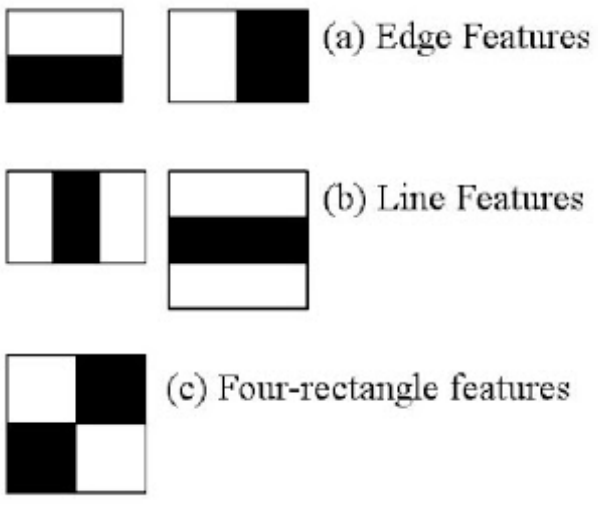

Figure 2.1: haar features[2]

features. (Just imagine how much computation it needs? Even a 24x24 window

results over 160000 features). For each feature calculation, we need to find sum of pixels under white and black rectangles. To solve this, they introduced the integral images. It simplifies calculation of sum of pixels, how large may be the number of pixels, to an operation involving just four pixels. Nice, isn't it? It makes things super-fast. But among all these features we calculated, most of them are irrelevant. For example, consider the image below. Top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applying on cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.

2.3.2 Boosting

Boosting is a machine learning ensemble . a family of machine learning algorithms which convert weak learners to strong ones. Boosting is based on the question Can a set of weak learners create a single strong learner?. A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing). In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.

How Boosting is done?

We apply each and every feature on all the training images. For each feature, it finds the best threshold (Which have minimum error rate) which will classify the faces to positive and negative. Final classifier is a weighted sum of these weak classifiers. It is

called weak because it alone can't classify the image, but together with others forms a strong classifier. In an image, most of the image region is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot. Don't process it again. Instead focus on region where there can be a face. This way, we can find more time to check a possible face region. For this they introduced the concept of Cascade of Classifiers

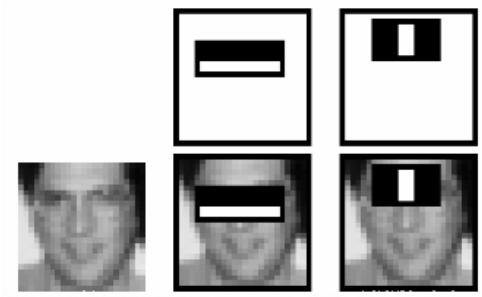

Figure 2.2: Feature Kernel Calculation[4]

2.3.3 Cascade of Classifiers

Instead of applying all the 6000 features on a window, group the features into different stages of classifiers and apply one-by-one. (Normally first few stages will contain very less number of features). If a window fails the first stage, discard it. We don't consider remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region. How is the plan !!! Authors' detector had 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in first five stages. (Two features in the above image is actually obtained as the best two features from Adaboost). According to authors, on an average, 10 features out of 6000+ are evaluated per sub-window

2.4 Face Recognition

There exist several methods for extracting the most useful features from (preprocessed) face images to perform face recognition. One of these feature extraction methods is the Local Binary Pattern (LBP) method. This relative new approach was introduced in 1996 by Ojala et al. [5]. With LBP it is possible to describe the texture and shape of a digital image. This is done by dividing an image into several small regions from which the features are extracted These features consist of binary patterns that describe the surroundings of pixels in the regions The obtained features from the regions are concatenated into a single feature histogram, which forms a representation of the image. Images can then be compared by measuring the similarity (distance) between their histograms. According to several studies [2, 3, 4] face recognition using the LBP method provides very good results, both in terms of speed and discrimination performance. Because of the way the texture and shape of images is described, the method seems to be quite robust against face images with different facial expressions, different lightening conditions, image rotation and aging of persons.

2.4.1 Principles of Local Binary Patterns

The original LBP operator was introduced by Ojala et al. This operator works with the eight neighbors of a pixel, using the value of this center pixel as a threshold. If a neighbor pixel has a higher gray value than the center pixel (or the the same gray value) than a one is assigned to that pixel, else it gets a zero.

The LBP code for the center pixel is then produced by concatenating the eight ones

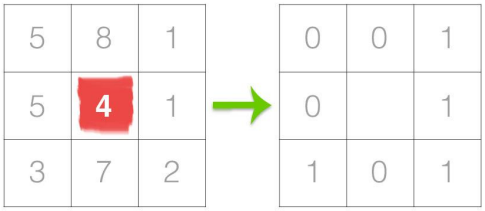

Figure 2.3: Thresholding of image[5]

or zeros to a binary code Later the LBP operator was extended to use neighbourhoods of different sizes. In this case a circle is made with radius R from the centre pixel. P sampling points on the edge of this circle are taken and compared with the value of the centre pixel. To get the values of all sampling points in the neighbourhood for any radius and any number of pixels, (bilinear) interpolation is necessary. For neighbourhood s the notation (P, R) is used illustrates three rates three neighbour-sets for different values of P and R. If the coordinates of the centre pixel are (xc, yc) then the coordinates of his P neighbours (xp, yp) on the edge of the circle with radius R can be calculated with the sinus and cosines:

$$xp = xc + Rcos(2p/P\ ) \quad (1)$$

$$yp = yc + Rsin(2p/P\ ) \quad (2)$$

If the gray value of the centre pixel is gc and the gray values of his neighbours are gp, with p = 0, ..., P 1, than the texture T in the local neighbourhood of pixel (xc, yc) can be defined as:

$$T = t(gc, g0, \ldots, gP) \quad (3)$$

Once these values of the points are obtained is it also possible do describe the texture in another way. This is done by subtracting the value of the center pixel from the

values of the points on the circle. On this way the local texture is represented as a joint distribution of the value of the center pixel and the differences:

$$T = t(gc, g0\ Jc, \ldots, gP\ Jc) \quad (4)$$

Since t(gc) describes the overall luminance of an image, which is unrelated to the local image texture, it does not provide useful information for texture analysis. Therefore, much of the information about the textural characteristics in the original joint distribution (Eq. 3) is preserved in the joint difference distribution (Ojala et al. 2001):

$$T\ (g0\ Jc, \ldots, gP\ Jc) \quad (5)$$

Although invariant against gray scale shifts, the differences are affected by scaling. To achieve invariance with respect to any monotonic transformation of the gray scale, only the signs of the differences are considered. This means that in the case a point on the circle has a higher gray value than the center pixel (or the same value), a one is assigned to that point, and else it gets a zero:

$$T\ (s(g0\ Jc), \ldots, s(gP\ gc) \quad (6)$$

Where In the last step to produce the LBP for pixel (xc, yc) a binomial weight 2p is assigned to each sign s(gp gc). These binomial weights are summed:

The Local Binary Pattern characterizes the local image texture around (xc, yc). The original LBP operator in figure 1.3 is very similar to this operator with P = 8 and R = 1, thus LBP8,1. The main difference between these operators is that in LBP8,1 the pixels first need to be interpolated to get the values of the points on the

circle. V. U LBP Feature Vector

The LBP feature vector, in its simplest form, is created in the following manner:

Divide the examined window into cells (e.g. 16x16 pixels for each cell).

For each pixel in a cell, compare the pixel to each of its 8 neighbours (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.

Where the centre pixel's value is greater than the neighbour's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).

Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the centre). This histogram can be seen as a 256-dimensional feature vector.

Optionally normalize the histogram.

Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window. The feature vector can now be processed using the Support vector machine or some other machine-learning algorithm to classify images. Such classifiers can be used for face recognition or texture analysis. A useful extension to the original operator is the so-called uniform pattern[8], which can be used to reduce the length of the feature vector and implement a simple rotation invariant descriptor. This idea is motivated by the fact that some binary patterns occur more commonly in texture images than others. A local binary pattern is called uniform if the binary pattern contains at most two 0-1 or 1-0 transitions. For example, 00010000(2 transitions) is a uniform pattern, 01010100(6 transitions) is not. In the computation of the LBP

histogram, the histogram has a separate bin for every uniform pattern, and all non-uniform patterns are assigned to a single bin. Using uniform patterns, the length of the feature vector for a single cell reduces from 256 to 59.

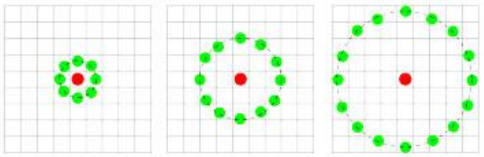

Figure 2.4: LBP Calculations for feature vector[6]

2.4.2 Creation of Image Database

The administrator of the house will create ids of the people and create a database of the image and create the training set of the algorithm . The implementation of the algorithm will be discussed later in this book.

Chapter 3

Communication Module and NLP

In this chapter, we discuss about the sub problems Communication Module and Natural Language Processing Module. Firstly, we understand about server-client paradigm and try to implement it over devices- Mobile, Arduino, OS.Next, we talk about the giving power of understanding the natural language and giving reply to the questions and converting those commands to the actions to the Operating system.

3.1 Introduction

Devices Mobile application and Arduino act as clients and OS acts as the server. The clientserver model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called client. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. Client-host and server-host have subtly different meanings

than client and server. A host is any computer connected to a network. Whereas the words server and client may refer either to a computer or to a computer program, server-host and user-host always refer to computers. The host is a versatile, multi-function computer; clients and servers are just programs that run on a host. In the clientserver model, a server is more likely to be devoted to the task of serving.

This chapter is divided into three parts:-

1. Android application-OS communication
2. OS-Arduino communication
3. Android application-OS communication
4. Connection

3.2 Android app and OS communication

Android app and OS communication is established using sockets.A socket is a software endpoint that establishes bidirectional communication between a server program and one or more client programs.Normally, a server runs on a specific computer and has a socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request. Both server(OS) and client(Mobile application, Arduino board) are connected over a local area network. We used IEEE 802.11 protocol.

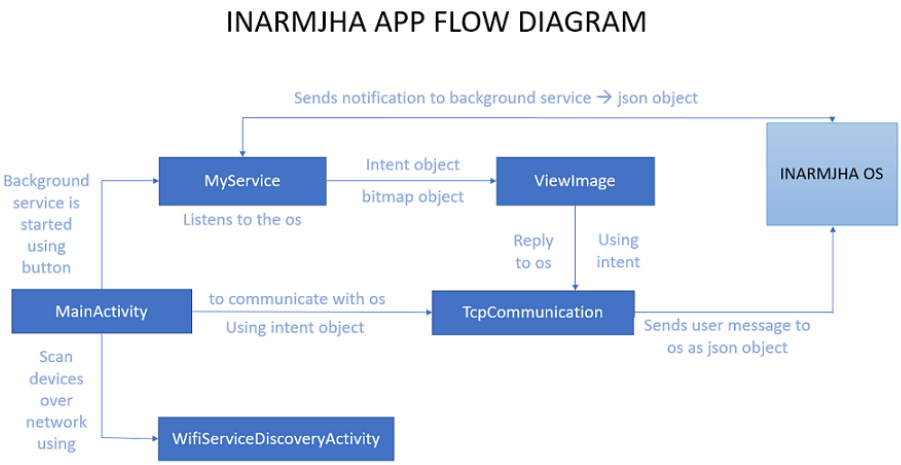

Figure 3.1: Flow Diagram of Android app

3.2.1 Client

The client knows the hostname of the machine on which the server is running and the port number on which the server is listening. To make a connection request, the client tries to rendezvous with the server on the server's machine and port. The client also needs to identify itself to the server so it binds to a local port number that it will use during this connection.

If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to the same local port and also has its remote endpoint set to the address and port of the client. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client.If everything goes well, the server accepts the connection. Upon acceptance, the server gets a new socket bound to the same local port and

also has its remote endpoint set to the address and port of the client. It needs a new socket so that it can continue to listen to the original socket for connection requests while tending to the needs of the connected client.The client and server can now communicate by writing to or reading from their sockets. An endpoint is a combination of an IP address and a port number. Every TCP connection can be uniquely identified by its two endpoints. That way you can have multiple connections between your host and the server. The java.net package in the Java platform provides a class, Socket, that implements one side of a two-way connection between your Java program and another program on the network. The Socket class sits on top of a platform-dependent implementation, hiding the details of any particular system from your Java program. By using the java.net.Socket class instead of relying on native code, your Java programs can communicate over the network in a platform-independent fashion.java.net includes the ServerSocket class, which implements a socket that servers can use to listen for and accept connections to clients. User connects to OS using its IP address generated over same network and port number on which the server program on OS is running through sockets by creating a new thread. To connect to different operating systems we use android shared preferences which store the IP address and port number given by the user and connect to that particular operating system.

3.3 Communication

After user establishes connection with the OS, he need to communicate with the OS. Key terms used in this sub module are:

1)Android Notification

2)JSON Objects

3) Base64 Encoding and Decoding

4)Android Bitmap (If message is a image)

Each message exchanged between server and client undergoes above steps:

3.3.1 Android Notification

A notification is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the notification area. To see the details of the notification, the user opens the notification drawer. Both the notification area and the notification drawer are system-controlled areas that the user can view at any time.

3.3.2 JSON Objects

JSON stands for JavaScript Object Notation.It is an independent data exchange format and is the best alternative for XML. For parsing a JSON object, we will create an object of class JSONObject and specify a string containing JSON data to it.

3.3.3 Base64 Encoding and Decoding

Base64 is a generic term for a number of similar encoding schemes that encode binary data by treating it numerically and translating it into a base 64 representation. The

Base64 term originates from a specific MIME content transfer encoding. Base64 encoding schemes are commonly used when there is a need to encode binary data that needs be stored and transferred over media that are designed to deal with textual data. This is to ensure that the data remains intact without modification during transport. The particular choice of characters to make up the 64 characters required for base varies between implementations. The general rule is to choose a set of 64 characters that is both part of a subset common to most encodings, and also printable. This combination leaves the data unlikely to be modified in transit through systems, such as email, which were traditionally not 8-bit clean. For example, MIME's Base64 implementation uses A-Z, a-z, and 0-9 for the first 62 values. Other variations, usually derived from Base64, share this property but differ in the symbols chosen for the last two values; an example is UTF-7.

3.3.4 Android Bitmap

Android provides Bitmap class to handle images. This can be found under android.graphics.bitmap. An image is nothing but a two dimensional matrix. Same way you will handle a bitmap. An image consist of pixels. So you will get pixels from this bitmap and apply processing to it. User receives a notification from OS, whenever a face is recognized by face recognition module in front of a door. User can reply a message to OS. User message is a string which is encoded into Base64 Format and sent to the OS in the form of JSON Object using Buffered Writer java class. OS receives the encoded message using Buffered Reader java class. It deceodes it and the JSON object converted it into a string. If the message to be sent is an image, its converted into a bitmap object and then into a string. String is converted into JSON

object. This message encoded using Base64 and sent to the app. App decodes the string and conevrts it into a bitmap object and shows it to the user.

3.4 Communication–The Work Flow

User receives a notification from OS, whenever a face is recognized by face recognition module in front of a door. User can reply a message to OS. User message is a string which is encoded into Base64 Format and sent to the OS in the form of JSON Object using Buffered Writer java class. OS receives the encoded message using Buffered Reader java class. It decodes it and the JSON object converted it into a string. If the message to be sent is an image, its converted into a bitmap object and then into a string. String is converted into JSON object. This message encoded using Base64 and sent to the app. App decodes the string and converts it into a bitmap object and shows it to the user and same process is done while server sends message to the user.

3.5 OS-Arduino communication

As discussed in the introduction section, we have used the Arduino UNO. To enable access to the network, Arduino should be connected with an Arduino ethernet shield.

3.5.1 Arduino Ethernet Shield

The Arduino Ethernet Shield allows an Arduino board to connect to the internet. It is based on the Wiznet W5100/W5200 ethernet chip providing a network (IP) stack capable of both TCP and UDP. Use the Ethernet library to write sketches which

connect to the internet via a RJ45 Ethernet jack.

Here lists the comparison among various versions of Ethernet Shield:

Parameter	Ethernet Shield V1.0	Ethernet Shield V2.0
Working Voltage	+5V	+5V
Control Mode	Control by Arduino	Control by seeeduino/Arduino
TCP/IP Ethernet	Controller	Wiz5100 Wiz5200
power supply mode	5V pin of Arduino/Seeeduino	5V pin of Arduino/Seeeduino
Standard Shield	No	Yes
SD Card Socket	No	Micro SD card in FAT16 or FAT32
Grove Connector	No	UARTandIICConnector
Ethernet jack	Standard RJ45	Minimal RJ45

Table summering the comparisions of various ethernet shields

The Arduino-OS communication is same as android app-OS communication but instead off using JSON objects we use xml tree.

3.6 Natural Language Module

The main purpose of this module is to make the operating system understand user commands and reply and act.

3.6.1 Natural Language Processing

Natural language processing (NLP) is a field of computer science, artificial intelligence and computational linguistics concerned with the interactions between computers and human (natural) languages, and, in particular, concerned with programming computers to fruitfully process large natural language corpora. We are trying to implement

the information retrieval model of the natural language. Information Retrieval means answering a question from a pre described question and answer pair. To implement this we use the machine learning API chatter bot.

3.6.2 Chatter bot

The Chatter bot API uses the machine learning algorithms for selecting a reply it just uses the voting method as user responds positively to a reply it increases the weight of the response and the highest weighted response will be replied to the user by using communication module.. The data set implemented was filtered from the twitter data set to contain only data that is applicable to the context of house hold personal assistant. To chat with the operating system the user can send a message from the mobile and the operating receives the message as described in the communication module and process it and sends the reply again as described in the communication module. A user can also send commands to the operating system include on Tv1,Turn on Washing machine etc. The operating System with the help of regular expressions tries to search for the keywords like on,off etc and send it to the devices module for further processing.

Chapter 4

Software Stack

A software stack is a group of programs that work in tandem to produce a result or achieve a common goal. Software stack also refers to any set of applications that works in a specific and defined order towards a common goal or any group of utilities or routine applications that works as a set. The software stack used in this project is

4.1 Android

4.1.1 History

Operating Systems have developed a lot in last 15 years. Starting from black and white phones to recent smart phones or mini computers, mobile OS has come far away. Especially for smart phones, Mobile OS has greatly evolved from Palm OS in 1996 to Windows pocket PC in 2000 then to Blackberry OS and Android. One of the most widely used mobile OS these days is ANDROID. Android is a software bunch comprising not only operating system but also middleware and key applications. Android Inc was founded in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in

2005. After original release there have been number of updates in the original version of Android.

4.1.2 Version

Requirements: Android mobile with API level higher than 19 or Kitkat (4.4) version[7] Target Audience 87.4

Code name	Version number	Initial release date	API level
(No codename)	1.0	September 23, 2008	1
Internally known as "Petit Four"	1.1	february 9, 2009	2
Cupcake	1.5	April 27, 2009	3
Donut	1.6	September 15, 2009	4
Eclair	2.02.1	October 26, 2009	57
Froyo	2.22.2.3	May 20, 2010	8
Gingerbread	2.32.3.7	December 6, 2010	910
Honeycomb	3.03.2.6	February 22, 2011	1113
Ice Cream Sandwich	4.04.0.4	October 18, 2011	1415
Jelly Bean	4.14.3.1	July 9, 2012	1618
KitKat	4.44.4.4	October 31, 2013	19
Lollipop	5.05.1.1	November 12, 2014	2122
Marshmallow	6.06.0.1	October 5, 2015	23
Nougat	7.07.1.2	August 22, 2016	2425

4.1.3 Android Software Stack

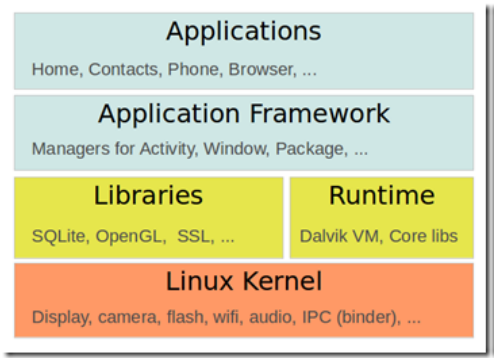

Figure 4.1: Android Software Stack[8]

4.1.4 Applications

These are the basics of Android applications:

1 Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers)

2 Each component performs a different role in the overall application behavior, and each one can be activated individually (even by other applications)

3 The manifest file must declare all components in the application and should also declare all application requirements, such as the minimum version of Android required and any hardware configurations required

4 Non-code application resources (images, strings, layout files, etc.) should include alternatives for different device configurations (such as different strings for different languages) To implement android we use java programming language.

4.2 Python

Socket programming is the heart of the client server communication we should be implementing with a programming language that has a wide range of features and python is one such. Python provides a module or library socket which provides a huge range of implementations for Tcp,Udp socket communications and python also provides libraries for the implementation of json and xml tree .Python also provides library for multi threading which is useful in developing a server.

4.3 OPEN CV

OpenCV[2] (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. The library is cross-platform and free for use under the open-source BSD license. OpenCV's application areas include:

1 2D and 3D feature toolkits

2 Egomotion estimation

3 Facial recognition system

4 Gesture recognition

5 Humancomputer interaction (HCI)

6 Mobile robotics

7 Motion understanding

8 Object identification

9 Segmentation and recognition

10 Stereopsis stereo vision: depth perception from 2 cameras

11 Structure from motion (SFM)

12 Motion tracking

13 Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

1 Boosting

2 Decision tree learning

3 Gradient boosting trees

4 Expectation-maximization algorithm

5 k-nearest neighbor algorithm

6 Naive Bayes classifier

7 Artificial neural networks

8 Random forest

9 Support vector machine (SVM) Open cv provides a rich interface with python programming language and also it supports a wide variety of operating systems and hardware processor implementations.

4.4 Chatter bot

ChatterBot is a Python library that makes it easy to generate automated responses to a users input. ChatterBot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users. For more details about the ideas and concepts behind ChatterBot see the process flow diagram. An example of typical input would be something like this:

user:Good morning! How are you doing?

bot: I am doing very well, thank you for asking.

user: You're welcome.

bot: Do you like hats?

4.4.1 Language Independence

The language independent design of ChatterBot allows it to be trained to speak any language. Additionally, the machine-learning nature of ChatterBot allows an agent instance to improve its own knowledge of possible responses as it interacts with humans and other sources of informative data.

4.4.2 How Chatter Bot Works

ChatterBot is a Python library designed to make it easy to create software that can engage in conversation. An untrained instance of ChatterBot starts off with no knowledge of how to communicate. Each time a user enters a statement, the library saves the text that they entered and the text that the statement was in response to. As ChatterBot receives more input the number of responses that it can reply and the accuracy of each response in relation to the input statement increase. The program selects the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

4.5 Arduino

Audrino is a open source software that provides to write and upload code for the microcontroller Arduino .It provides a rich environment for developing codes for Arduino with programming languages like c++,Python etc..

4.6 Operating System Stack

Operating System Stack deals with the package distributions used in building the operating system.This mainly consists of:-

1)Virtual Box

2)Arch Linux

3) Gnome Desktop Environment

4.6.1 Oracle Virtual Box

A VirtualBox is a software virtualization package that installs on an operating system as an application. VirtualBox allows additional operating systems to be installed on it, as a Guest OS, and run ina virtual environment.

4.6.2 Arch Linux

Arch Linux is an independently developed, x86-64 general-purpose Linux distribution that strives to provide the latest stable versions of most software by following a rolling-release model. The default installation is a minimal base system, configured by the user to only add what is purposely required. ArchLinux kernel version is 4.9.6-1-ARCH. The pacman is the package manager of Arch Linux. It combines a

simple binary package format with an easy-to-use build system. The goal of pacman is to make it possible to easily manage packages, whether they are from the official repositories . pacman keeps the system up to date by synchronizing package lists with the master server. This server/client model also allows the user to download/install packages with a simple command, complete with all required dependencies. pacman is written in the C programming language and uses the .pkg.tar.xz package format.

4.6.3 GNOME Desktop Environment

Arch Linux comes without GUI .So GNOME Desktop environment is installed using pacman. The GNOME desktop is a free, user-friendly desktop environment for Unix and Linux operating systems. It also provides a platform for development of new applications. GNOME version used is GNOME Sell 3.22.2 Two groups are available:

Two groups are available:

1.gnome contains the base GNOME desktop and a subset of well-integrated applications

2.gnome-extra contains further GNOME applications, including an archive manager, disk manager, text editor, and a set of games. Note that this group builds on the gnome group.

Chapter 5

Design and Implementation

In the previous chapters we have introduced our objective and introduced all the theoretical concepts. In this chapter we are going to discuss the design and implementation of the project.

5.1 Operating System Services

Arch Linux Kernel is used.The linux distribution is installed in oracle virtual box. Required packages,like firefox browser,Java,GTK themes,media players,etc.. are added using package manager pacman.By default Arch linux has Command Line Interface(CLI).It has no GUI. So, GNOME desktop is installed. We designed our user interface by making changes in .css file and .xml file in the gnome-shell theme directory. The .css file can be changed according to our requirements.

Figure 5.1: Login Page

5.1.1 Background Service

We start the services as daemons(background processes). systemd[9] is the default init framework. The services which are started by systemd can be found in the subfolders of /etc/systemd/system/. Services can be enabled using the systemctl command.It provides a system and service manager that runs as PID 1 and starts the rest of the system. The services are defined using configuration files called unit files[10].

JHA provides autostart process Face Detection, chatter bot API, using the unit files.

Unit file looks like this:

[Unit]

Description=Face Detection Service

After=multi-user.target

Type=idle

ExecStart=/usr/bin/python/home/inarmjha/Downloads/facerecognition.py

stall

WantedBy=multi-user.target

Figure 5.2: Desktop Menu

5.2 Android Application

Android application receives a notification from the OS, whenever face recognition module finds an human in front of the door camera.

User can view the image by clicking on the notification and can reply to OS.

The home screen of the app has option to start connection with OS,disconnect with the OS,send message to the OS and scan the devices over a network(future work).

User has an option of either enabling or disabling the connection with OS which

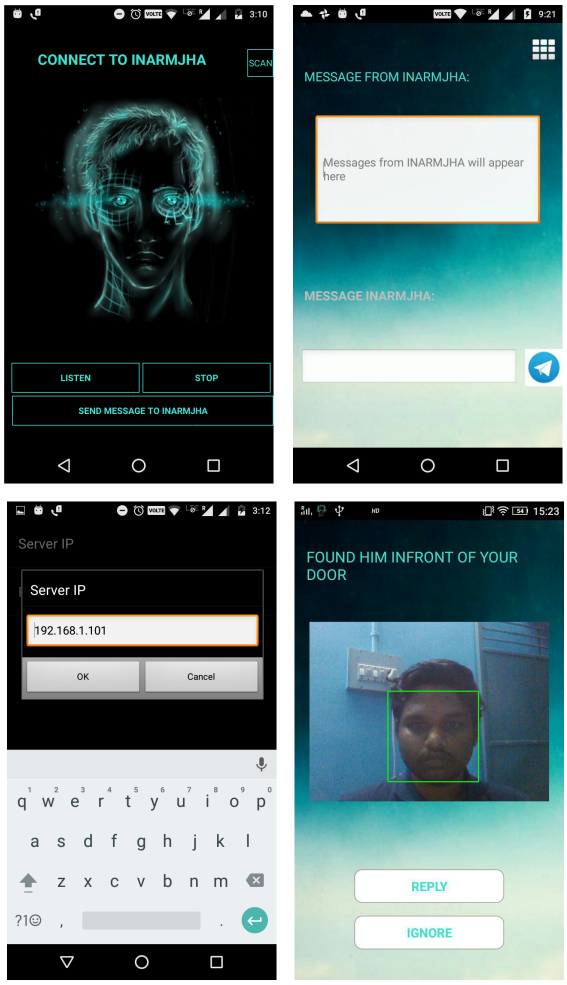

Figure 5.3: Android App Interface

runs as an background android service.User can click LISTEN button to start connection with the OS or can quit connection by clicking STOP button.

User has an option of connecting to different OS(not at a time) by changing the server IP and port number in the settings section of the application.

Thus, user can connect to the operating system and start communication with the OS to control smart devices of a home.

Chapter 6

Intelligent Door Locking System

In this chapter we are going to discuss the design and implementation of the door locking system to provide security to the house.In the previous chapters we have described various aspects regarding the software components of the project.In this chapter we will be going into a deep discussion into the hardware details of the motor implementations etc.

6.1 Introduction

The main motive of this Intelligent system is to provide security to the user by implementing a face detection algorithm as described in previous chapters.Once the face has been detected the operating system sends a image to the mobile of the user as a notification.If the user replies the operating system with the command open then the operating system sends the command to the ardino which is the brain of the intelligent Door locking system.Now,we will explore how we have designed the Locking System.

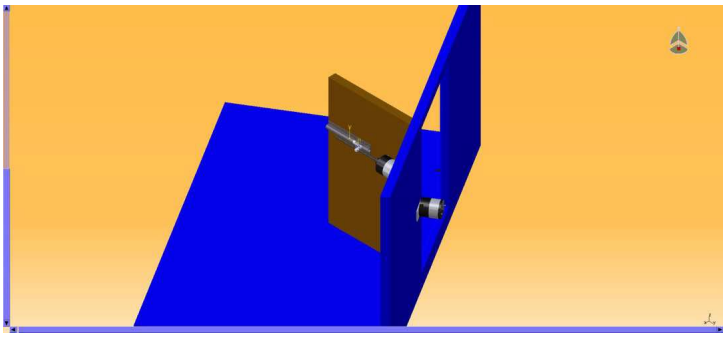

Figure 6.1: Door Backend

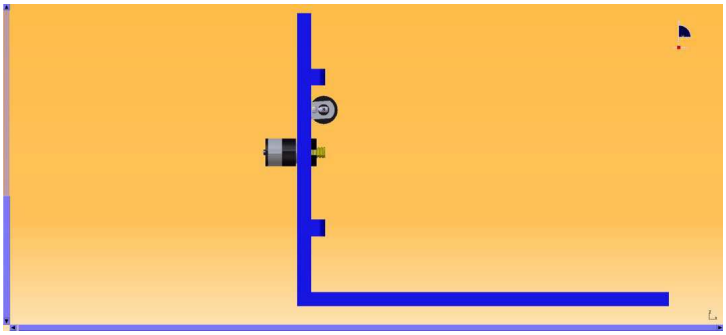

Figure 6.2: Door Side View

6.2 Door Locking System

The Door Locking system is been implemented as a two step process.

1 Latch Mechanism

2 Actuation for Door Movement

6.2.1 Latch Mechanism

Latch Mechanism is mainly based on Nut and Bolt Principle.The main components of the latch are

1 two cylindrical casings

2 Cylindrical Rode

The Motion has to be provided to the cylindrical tube inorder to engage coaxially with the two cylindrical casings.The Motions is provided with the help of nut and bolt principle.Rotatory motion of the bolt is provided with the help of DC Motor which provides linear coaxial motion for the cylindrical rod which is rigidly attached with the nut.In this,Rotatory motion of the motor is converted into linear coaxial motion of the cylindrical rod.

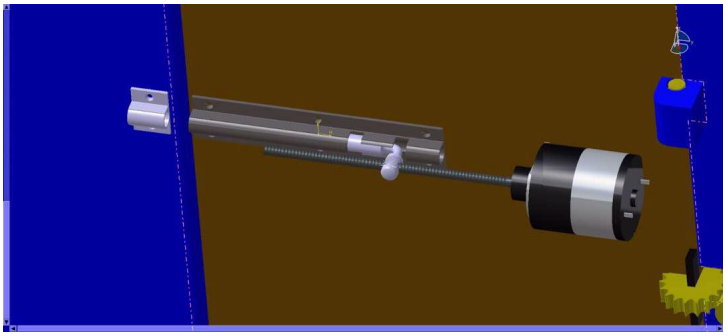

Figure 6.3: Latch

6.2.2 Actuation For Door Movement

An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system. An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power. The supplied main energy source may be electric current, hydraulic fluid pressure, or pneumatic pressure. When the control signal is received, the actuator responds by converting the energy into mechanical motion. An actuator is the mechanism by which a control system acts

upon an environment. The control system can be simple (a fixed mechanical or electronic system), software-based (e.g. a printer driver, robot control system), a human, or any other input. The movement that has to be provided for the door is a rotatory motion about the hinged access of the door mechanism.Inorder to provide this we use worm gear/wheel mechanism.

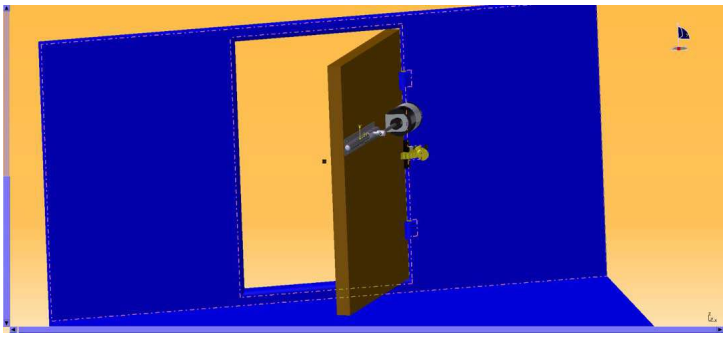

Figure 6.4: Door locking Mechanism

6.2.3 Worm Gear/Wheel Mechanism

Worm gear is a mechanical arrangement consisting of a toothed wheel worked by a short revolving cylinder (worm) bearing a screw thread.This has to be employed to efficient control of the movement and to provide higher torque requirement.

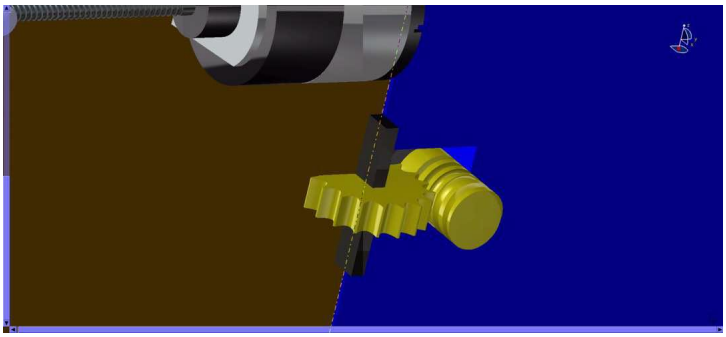

Figure 6.5: Gear

Chapter 7

Conclusion and Future Work

7.1 Conclusion

With this project, we helped humans no need to waste work on the activities where actually they need not and we successfully implemented a way to incorporate the artificial intelligence technology into the operating system.

We also successfully designed a Door automation system that automatically opens and closes the door on the command of the operating system.

7.2 Future Work

We had proposed a solution for communication between arduino and operating system still some quality of work has to be done to make it live. This project can be expanded to a wide range of domains and scales.Some of them to be included are :-

1) Behaviour Recognition system

2) Automatic Web Scaling Features

3) Intelligent Guesture and trojen Detection system

Bibliography

[1] Rapid Object Detection using a Boosted Cascade of Simple Features

[2] www.opencv-python-tutroals.readthedocs.io

[3] www.opencv-python-tutroals.readthedocs.io

[4] www.opencv-python-tutroals.readthedocs.io

[5] www.docs.opencv.org

[6] en.wikipedia.org/wiki/localbinarypatterns

[7] http://www.beginandroid.com/intro.shtml

[8] https://dzone.com/articles/android-software-stack-and

[9] https://wiki.archlinux.org/index.php/systemd

[10] www.raspberrypi-spy.co.uk