ROBO – AID: THE AUTONOMOUS FETCHING ROBOT

Project report submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

LAKSHEY JUNEJA (131028)

ANMOL MAHAJAN (131072)

Under the Supervision of

Mr. MUNISH SOOD

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

May 2017

TABLE OF CONTENTS

DECLARATION BY THE SCHOLARS

We hereby declare that the work reported in the B-Tech thesis entitled **"ROBO-AID"** submitted at **Jaypee University of Information Technology, Waknaghat, India,** is an authentic record of our work carried out under the supervision of **Mr. Munish Sood**.

We have not submitted this work elsewhere for any other degree or diploma.

LAKSHEY JUNEJA ANMOL MAHAJAN

Department of Electronics and Communication Engineering.

Jaypee University of Information Technology, Waknaghat, India.

02/05/2017

SUPERVISOR’S CERTIFICATE

This is to certify that the work reported in the B-Tech. final year project entitled “**ROBO-AID**”, submitted by **Sanna, Lakshey Juneja** and **Anmol Mahajan** at **Jaypee University of Information Technology, Waknaghat, India,** is a bonafide record of their original work carried out under my supervision.

This work has not been submitted elsewhere for any other degree or diploma.

____

Mr. MUNISH SOOD

Assistant Professor (Grade – II)

Department of Electronics and Communication Engineering

Jaypee University of Information Technology, Waknaghat.

ACKNOWLEDGMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this project. First and foremost, we would like to express our heartfelt thanks to our project guide and mentor **Mr. Munish Sood.**

His thinking and straight foreword attitude has inspired us to complete this project under stiff time limits. Thank you for the motivation and enlightening insights. Without your support and guidance, we would not have completed the first stage of our project. We are especially grateful that you treat us like a friend, respect our decisions and plans and encourage us with every progress.

Furthermore, we would also like to acknowledge with much appreciation the crucial role of the lab staff who gave the permission to use all required equipment and the necessary materials to complete the task.

DATE:

LAKSHEY JUNEJA

ANMOL MAHAJAN

LIST OF ABBREVIATIONS AND ACRONYMS

RPi 3	-	Raspberry Pi 3
DC	-	Direct Current
SSH	-	Secure Shell
GPIO	-	General Purpose Input/ Output
USB	-	Universal Standard Bus
BIOS	-	Basic Input/ Output System
RPM	-	Revolutions per minute
IC	-	Integrated Circuit
OS	-	Operating System

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

With the ever-rising use of machines we are seeing a shift in our lifestyles, moving form a more traditional to a western way of life. This has resulted in households that have almost all members working a job while the old members, kids and injured stay at home. There is also a shift to nuclear families so, there might be times when no one is available to take care of an injured, physically challenged or an old member of the family. Our project Robo-Aid is designed to cater to the needs of people in such scenarios.

Robo-Aid is an autonomous robot which serves the purpose of taking care of the Ill, Injured, old or physically challenged members of the household by bringing them articles that they need and are out of reach, or bringing medicines when required and such. The range of applications can be expanded to a helping robot for the kitchen, automated trolley for hospitals and so on.

CHAPTER 1

INTRODUCTION

Today, due to industrialization, we are seeing automation of many tasks in our day to day life and in the commercial and industrial areas as well. With the ever-rising use of machines we are also seeing a shift in our lifestyles, moving form a more traditional to a western way of life. This has resulted in households that have almost all members working a job while the old members, kids and injured stay at home. There is also a shift to nuclear families so, there might be times when no one is available to take care of an injured, physically challenged or an old member of the family.

This shift in lifestyle has many people depending on automated machines or devices for daily tasks and chores, like the prevalence of washing machine in every household compared to 10 years before, even more to automated washing machines that automatically start up when there’s water supply available every day at the same time. Similarly, automated floor cleaners are abundant in the market now too which are very useful for people that don’t have much time to clean up every day because of their hectic jobs and busy schedules. This rise of use and acceptance of their usefulness in our lives has enabled even greater penetration of these products in modern households.

Robo-Aid is an autonomous robot which serves the purpose of taking care of the Ill, Injured, old or physically challenged members of the household by bringing them articles that they need and are out of reach, or bringing medicines when required and such. The range of applications can be expanded to a helping robot for the kitchen, automated trolley for hospitals and so on.

1.1 WHAT IS THE PROJECT?

- Automated Bot for assisting physically injured/challenged people
- We plan on making a robot that helps a person who's not able to move around independently.
- The robot will be programmed to bring the articles to the person that they might need from any place in the room.
- For example, if a person has a fractured leg and is unable to retrieve the articles for him/herself. this robot will help in fetching those things for him/her.
- The injured person will give the bot a signal for bringing the article to him/her, the robot on getting the command, will start searching the room for the article and once located it’ll fetch the article.

1.2 SYSTEM DESIGN

The design of the project can be divided into different modules. They are :-

- Robot that can move in all directions
- A Raspberry Pi and Camera for Image processing
- Retrieval of the article detected

1.3 NEED FOR SUCH A SYSTEM

Such a system can be really helpful for a great number of tasks ranging from mundane tasks to specialized cases. Such existing spheres where an advanced application of the system can be instrumental are:

- Help for injured people who are unable to move or retrieve an object across the room on their own.
- Domestic helping robot in the kitchen that can carry around ingredients as per commands.
- Fully automated nursing robots that can deliver medicine to patients at allotted intervals.

CHAPTER 2

COMPONENTS AND TOOLS USED

2.1 PROJECT BOARD: RASPBERRY PI 3 MODEL B

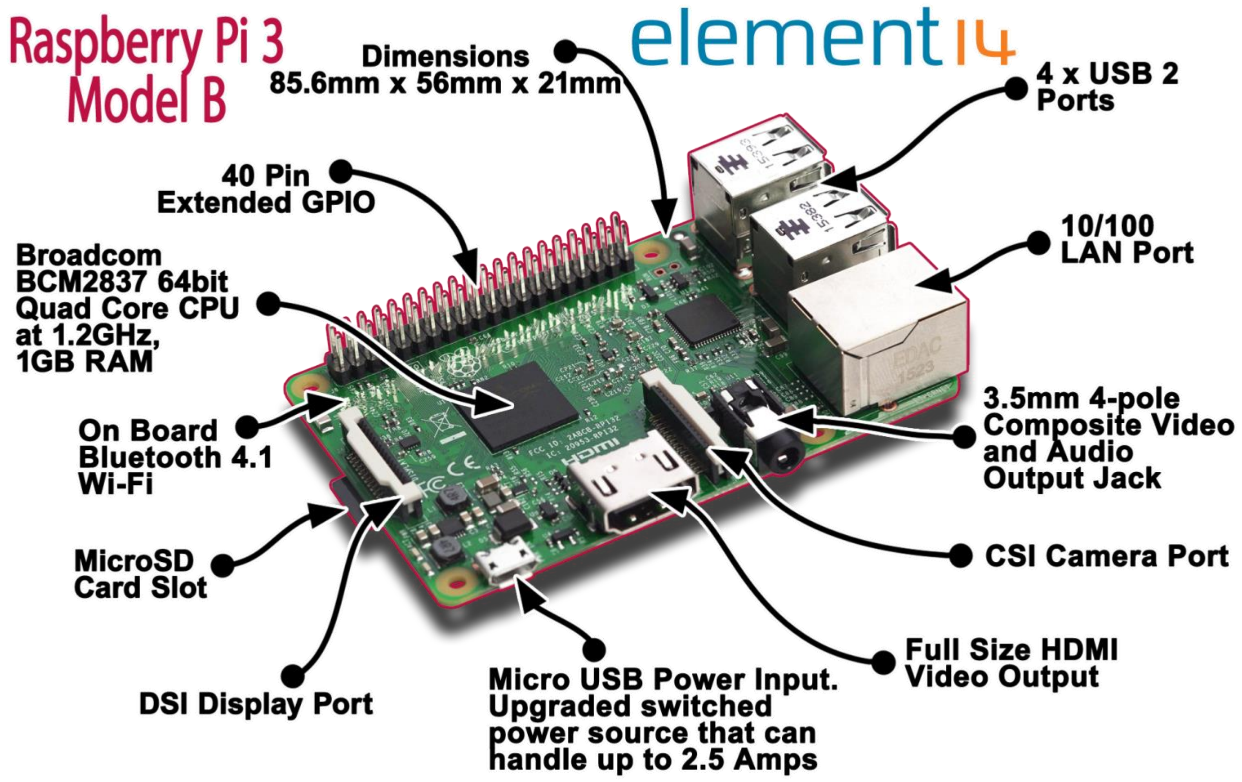
Raspberry Pi 3 Model B
element14
Dimensions 85.6mm x 56mm x 21mm
4 x USB 2 Ports
40 Pin Extended GPIO
10/100 LAN Port
Broadcom BCM2837 64bit Quad Core CPU at 1.2GHz, 1GB RAM
3.5mm 4-pole Composite Video and Audio Output Jack
On Board Bluetooth 4.1 Wi-Fi
CSI Camera Port
MicroSD Card Slot
Full Size HDMI Video Output
DSI Display Port
Micro USB Power Input. Upgraded switched power source that can handle up to 2.5 Amps

Figure 2.1: Raspberry Pi and components

2.1.1 Overview

Raspberry Pi 3 Model B is the most powerful Pi till date. It has a new 64-bit quad-core ARMv8 processor. The four ARM Cortex-A53 cores have a clock speed of 1.3 GHz.

Its features consist of integrated 802.11n Wi-Fi, Bluetooth 4.1 and Bluetooth Low Energy support which were absent in its predecessors. This makes more USB ports available to the user and reduces the need to connect external adapters for these functions.

It runs on a supply of 5V via a micro USB connection while drawing maximum power of 2.5A, making it quite power efficient.

The Pi 3 is designed closely to its predecessors in order to make the projects designed for older Raspberry Pi Model B to work in this model also, since all the ports and connectors have the same configuration. This also means most cases and mounting also remain consistent and backwards compatible through different iterations.

The Raspberry Pi 3 is the third-generation Raspberry Pi. Added features over the second-generation Pi are:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

Like the Pi 2, it also has:

- 1GB RAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot (now push-pull rather than push-push)
- VideoCore IV 3D graphics core

2.1.2 Specifications

SoC: Broadcom BCM2837

CPU: 4× ARM Cortex-A53, 1.2GHz

GPU: Broadcom VideoCore IV

RAM: 1GB LPDDR2 (900 MHz)

Networking: 10/100 Ethernet, 2.4GHz 802.11n wireless

Bluetooth: Bluetooth 4.1 Classic, Bluetooth Low Energy

Storage: MicroSD

GPIO: 40-pin header, populated

Ports: HDMI, 3.5mm analogue audio-video jack, 4× USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

2.1.3 Configuration

Similar to the second-generation Pi, it has a full-sized HDMI port, a 10/100 Ethernet port, four USB2 ports, a combined 3.5mm audio/composite video out port and a MicroSD card slot. The Broadcom VideoCore IV graphics chipset from the Pi 2 is the same as Pi 2 and the Pi 3 has 1GB of RAM.

The Pi 3 also has a 40-pin General Purpose Input/output connector, which allows us to access a lot of external devices and electronic components simultaneously for out project whilst leaving room for future upgrades to improve the presented prototype. It also consists of both camera and display connectors which use ribbon connectors.

It supports a wide range of operating systems. The recommend OS is Raspbian OS which is what we are using. It also supports Windows 10 IOT, if that is a preferred OS for those using a windows environment and a wide range of niche and specialized operating systems which can be used for development or as media centers.

In spite of being extremely popular the Pi 3, is still new and not all popularly used Raspberry Pi operating systems are fully optimized for it. For instance, the stable version of windows 10 IoT from last November is only designed for Pi 2, not for Pi 3 and we still don't have full Bluetooth support on ubuntu mate.

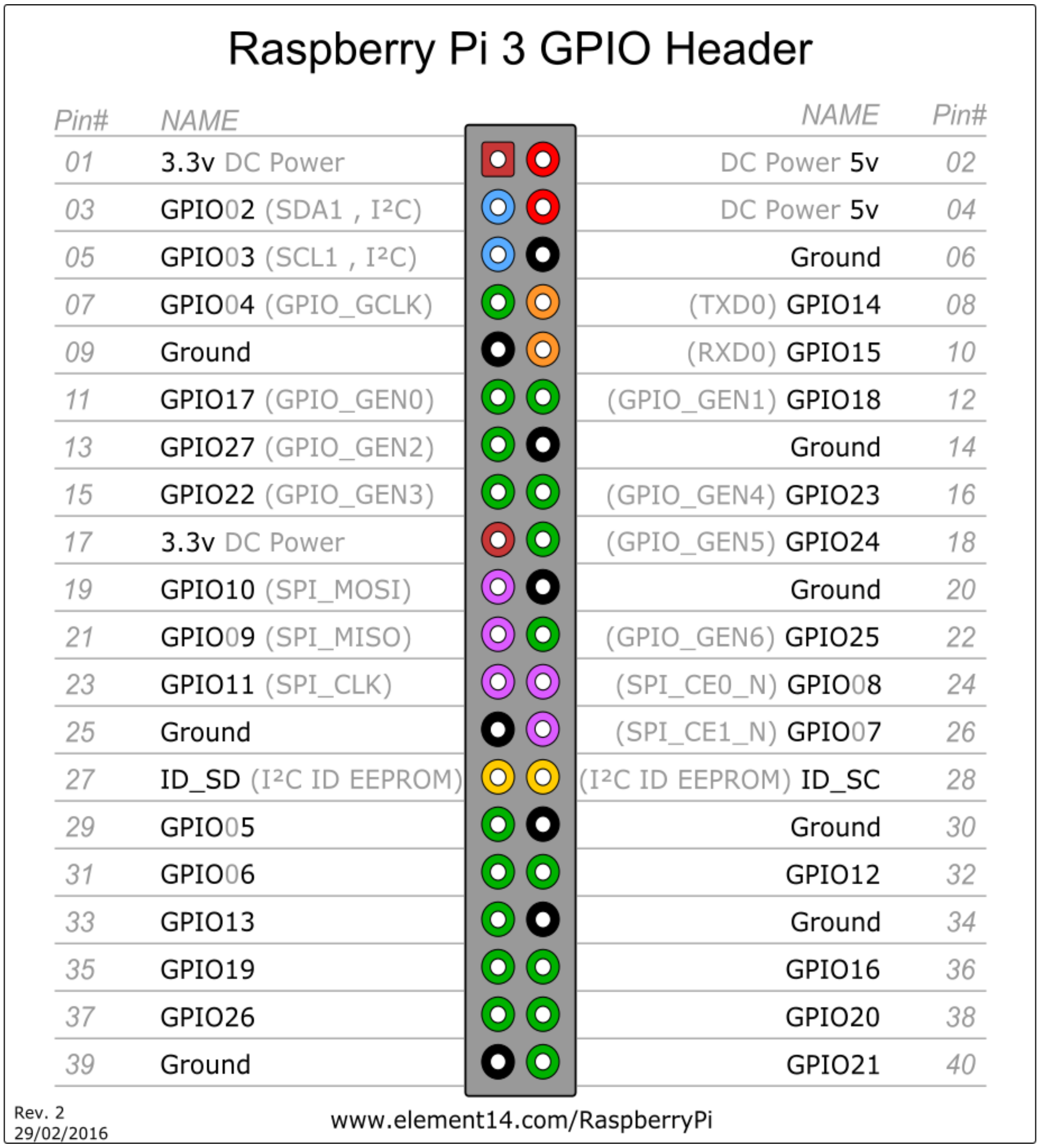

Figure 2.2: GPIO header

Like its siblings, the Raspberry Pi 3 doesn't have a BIOS. The Raspberry Pi stores its system configuration in “config.txt” file in the boot folder. Here you can set things like display properties, memory handling, enable camera support (for the camera port on the board) and overclock or underclock the CPU. Most of the stated features can be configured from within Raspbian which is the recommended OS, but not other operating system distros. This can be done using the raspi-conf utility. These boot settings are convenient if you don't want your Raspberry Pi to automatically detect the correct resolution without turning on your display before the computer boots.

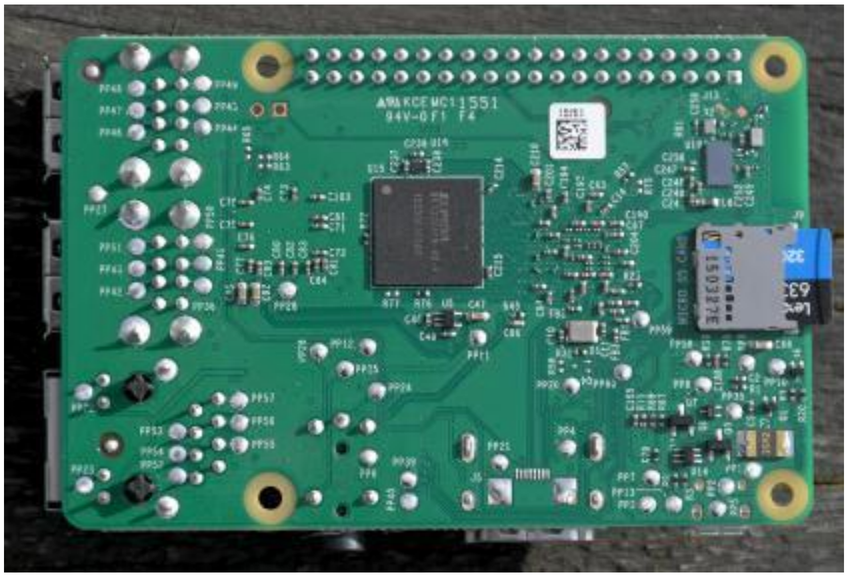

Figure 2.3: Physical Appearance of RPi 3

2.1.4 Performance

The Pi 3 is quite a bit ahead of previous Pi models in terms of performance thanks to the new upgraded quad-core processor. This makes it about 25% faster than the previous Pi iteration, i.e. Raspberry Pi 2 Model B.

The quad-core 1.2Ghz ARMv8 processor in the Pi 3 being very capable is able to give us a smooth desktop experience even when browsing the internet with multiple tabs. Albeit very capable, it is still not up to the task for heavy-duty multitasking, but we can open multiple browser tabs, edit documents and play media without any major slowdowns or hindrance to productivity.

2.1.5 Advantages

- Low price.
- Improved performance over previous generation.
- Integrated Wi-Fi 802.11n and Bluetooth 4.1 saves hassle and USB ports.
- Upgraded processor means smoother desktop performance.

2.1.6 Disadvantages

- Requires lots of additional hardware to function as a full PC.
- Limited operating system selection.
- Software setup may prove challenging.
- Slightly more power hungry than older Pis

2.1.7 Why did we use Raspberry Pi 3?

- Affordable and does not significantly increase the price of the final prototype.
- It is a really powerful computer so it gives us room to include more features.
- Because of the integrated Wi-Fi, it can work on any home Wi-Fi network and can be accessed easily with the help of any device connected to that home network. With fast processor, we are able to run image processing algorithms smoothly without any slowdowns or lag.

2.2 DC MOTORS

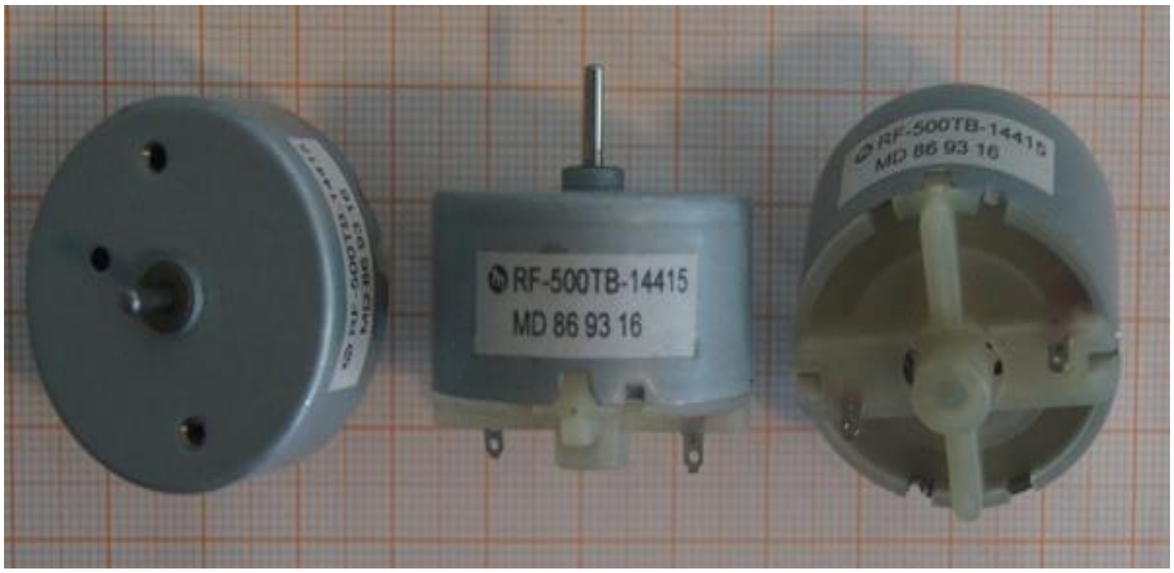

Figure 2.4: DC motor

A DC motor belongs to the class of electrical machines. It converts electrical energy to mechanical energy. DC Motors mostly rely on magnetic fields produced by currents for conversion.

A motor is an electrical machine which works on the principle that under the influence of a magnetic field, a current carrying conductor experiences mechanical force. Using Fleming's left hand rule, we can determine the direction of this force.

We have used 2 DC MOTORS for our project which run at 12V and 100 rpm.

2.3 IR SENSOR

An IR sensor essentially consists of a photodiode and an IR LED with a voltage comparator and suitable resistors to find the voltage change. The IR LED emits Infrared radiation and a photodiode is used to detect it.

Infrared radiation is not visible to the human eye and is produced by heat producing bodies. An infrared LED is powered by 3V, 20mA of power. A photodiode here works like a Light

Dependent Resistor (LDR) i.e. its impedance varies with the amount of radiation (Infrared radiation in this case) falling on it. It has a very high resistance when there is no radiation incident on it and this resistance falls when exposed to the radiation and starts conducting. It is used in reverse bias and looks like an LED with black coating on its outside.

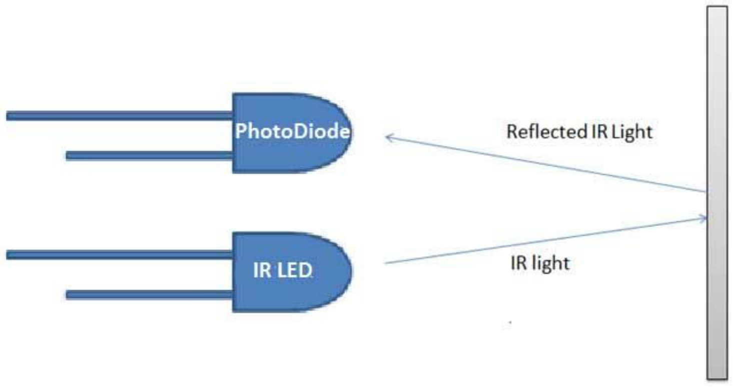

Figure 2.5: Working of IR Sensor

The voltage change across the photodiode is detected by a voltage comparator. An op-amp is usually used for this task, LM358 being the most common.

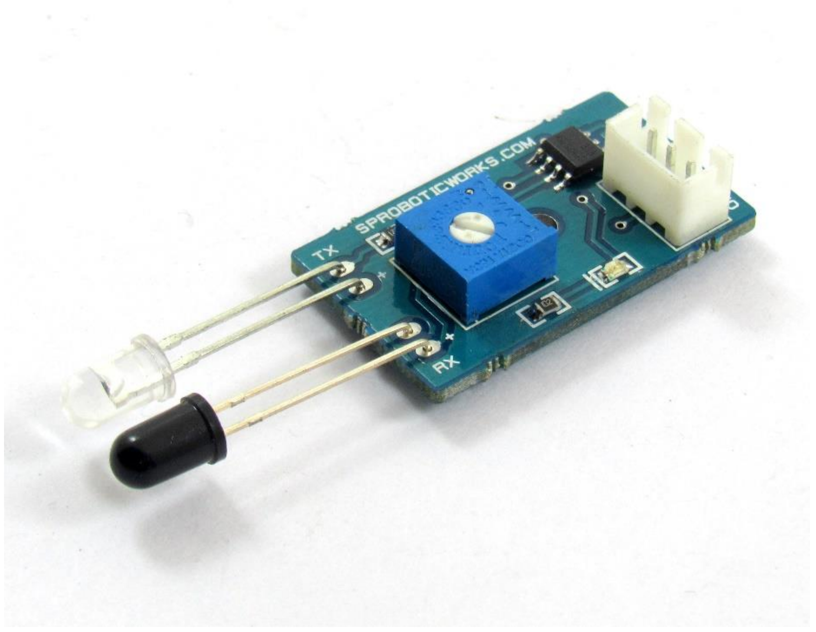

Figure 2.6: Practical IR Sensor

The photodiode and the LED are kept side to side. When an object is kept in front of the pair, the infrared radiation is reflected off the surface of the obstacle and reaches the photodiode. Hence, the resistance across it drops and it starts conducting. The comparator then detects the change in inputs and gives an output signal accordingly.

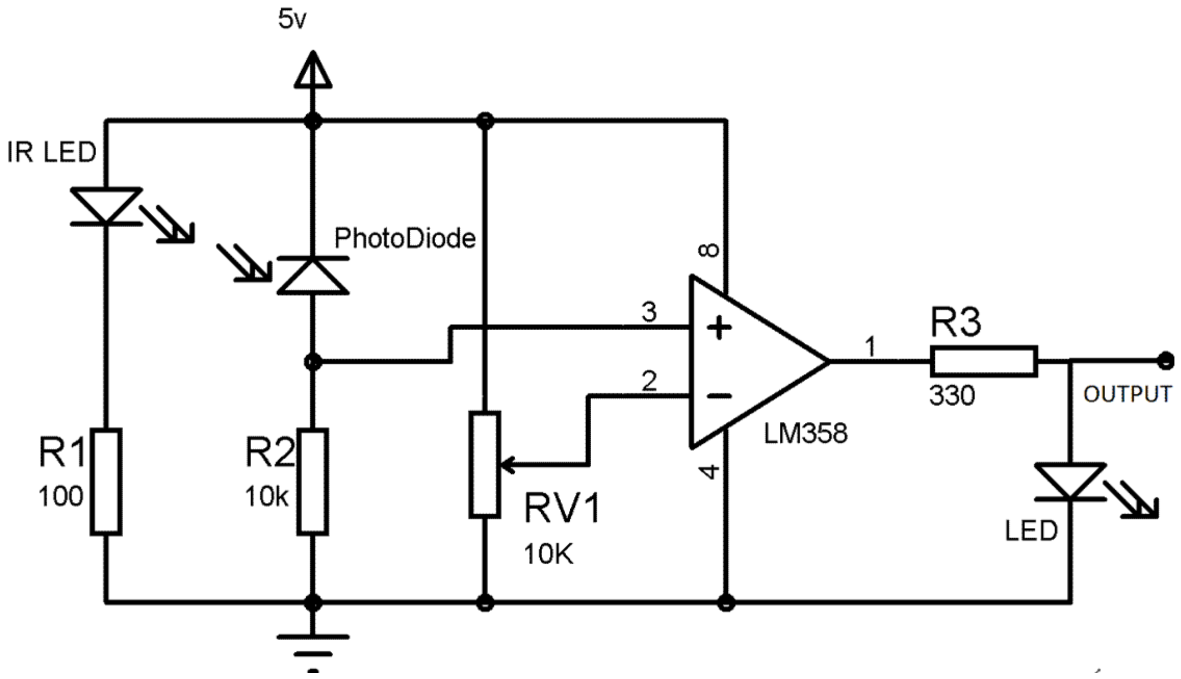

Figure 2.7: Circuit for IR Sensor

2.4 POWER SUPPLY

2.4.1 SUPPLY FOR RASPBERRY PI

We are using a 5V 2A power bank to power out Raspberry Pi on the go so that the robot is not bounded by any wires to the wall or otherwise.

2.4.2 12V BATTERY

To power our DC motors, we are using a 12V battery and using an L293D to control the power sent to the motors.

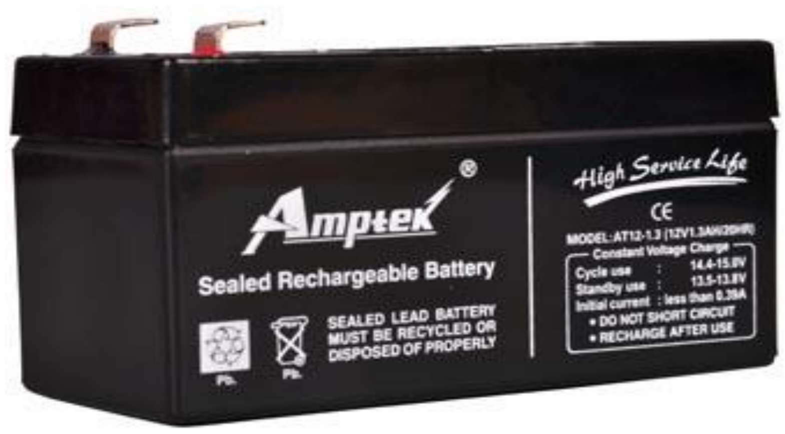
Amptek
Sealed Rechargeable Battery
SEALED LEAD BATTERY MUST BE RECYCLED OR DISPOSED OF PROPERLY
High Service Life
MODEL: AT12-1.3 (12V1.3AH(20HR)
Constant Voltage Charge
Cycle use : 14.4-15.0V
Standby use : 13.5-13.8V
Initial current : less than 0.39A
• DO NOT SHORT CIRCUIT
• RECHARGE AFTER USE

Figure 2.8: 12V Battery

MODEL NO:	AT12-1.3 (12V1.3AH)
WEIGHT:	0.57 KG
DIMENSIONS:	97 X 43 X 52 MM

2.5 METALCHASSIS

The design we have opted for here is a 3-legged design with 2 motors driving and one castor wheel in the front. This makes out bot very compact and space efficient.

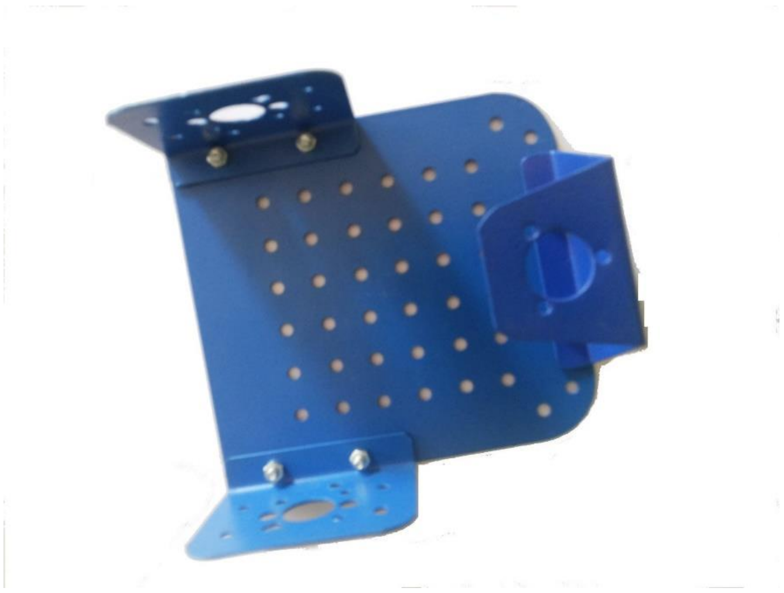

Figure 2.9: Metal Chassis

2.6 L293D IC

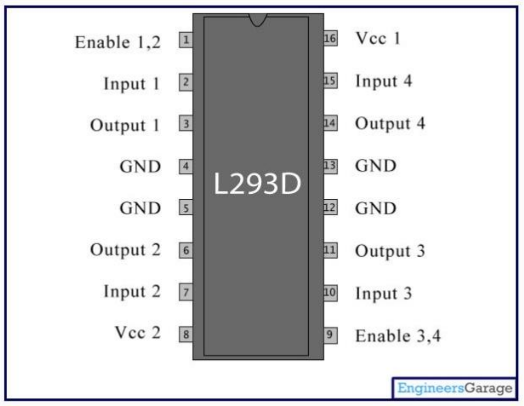

Figure 2.10: L293D Pinout

L293D is a Motor driver or Motor Driver IC which allows us to drive the DC motor to in either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. This means with one L293D, we can control 2 DC motors.

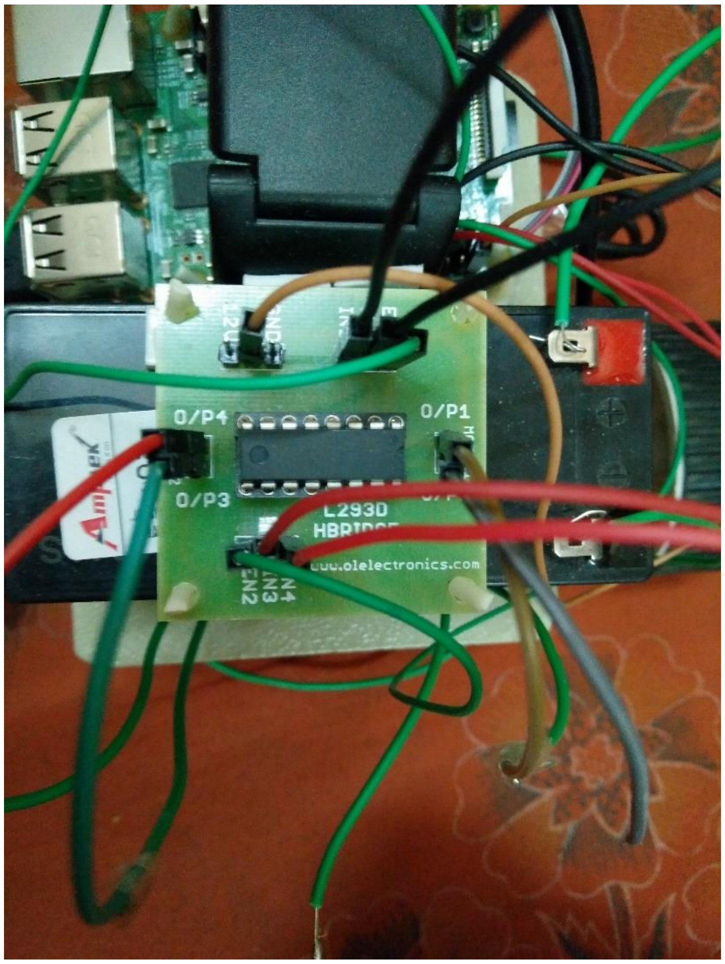

Figure 2.11: Appearance

VCC needs 5V input, it is the operational voltage of the IC; L293D will not use this voltage for driving the motor. VSS (V supply), is used to drive the motors which is 12V, and provided for with a different source than the rest of the components in this case. It means if we want to operate a motor at 9V then you need to provide a Supply of 9V across VSS Motor supply. The maximum voltage for VSS motor supply is 36V. It can supply a max current of 600mA per channel. Since it can drive motors Up to 36v hence you can drive pretty big motors with this simple IC.

Pin No	Function	Name
1	Enable pin for Motor 1; active high	Enable 1,2
2	Input 1 for Motor 1	Input 1
3	Output 1 for Motor 1	Output 1
4	Ground (0V)	Ground
5	Ground (0V)	Ground
6	Output 2 for Motor 1	Output 2
7	Input 2 for Motor 1	Input 2
8	Supply voltage for Motors; 9-12V (up to 36V)	Vcc_2
9	Enable pin for Motor 2; active high	Enable 3,4
10	Input 1 for Motor 1	Input 3
11	Output 1 for Motor 1	Output 3
12	Ground (0V)	Ground
13	Ground (0V)	Ground
14	Output 2 for Motor 1	Output 4
15	Input2 for Motor 1	Input 4
16	Supply voltage; 5V (up to 36V)	Vcc_1

TABLE 2.1: Pin Description of L293D

2.7 CAMERA

We are using the Logitech C170 webcam for our project that will help us with the image processing operations. It is relatively cheap and offers sufficient quality of images for our application given that our testing constraints are met.

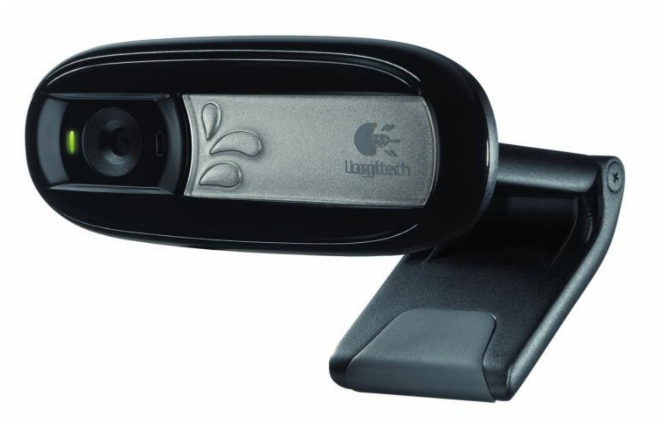

Figure 2.12: Logitech C170 webcam

2.8 OPENCV

2.8.1 Overview

OpenCV is a library of programming functions written in C++. It is mainly developed with a focus on real-time Computer Vision Applications. It supports a wide range of platforms including Windows, Linux, Mac OS, iOS and Android. It is Highly optimized code and can scale well on multi-core systems. It can also take advantage of hardware acceleration. The release of OpenCV is under a BSD license so it is totally free for use in academic and commercial applications.

2.8.2 History

Originally it was developed as a research initiative at an Intel research centre in Russia. Today, its development is being led by Itseez. The library was initially developed to increase efficiency of CPU heavy tasks. The first public release for OpenCV was in 2000 as an alpha version. The next major release was OpenCV 2.0 in 2009 adding more functions, better multi-core optimizations, alterations to the C++ interface and easier patterns. The releases not follow a biannual pattern.

2.8.3 Language Used

The OpenCV library is written with C++ as the language the source code is written in. However, there is comprehensive support for Python, Java and MATLAB with dedicated bindings. Extensive Documentation can be found online on these. Any new development and code being written in in the C++ interface

OpenCV being open source, welcomes any and all user contributions. Users can implement new functions, fix bugs or even report them and can request to update the future builds.

2.9 PYTHON

2.9.1 Overview

The programming language used in this project is Python. It is a high-level language for programming. Its syntax is relatively much simpler and takes lesser line of code to write a program as compared to C++.

It has a large library and extensive documentation which can be used to comprehend the working of the language. Code written in Python is easily readable and uses whitespace to define code blocks instead of using curly braces in traditional languages like C, C++, Java etc. Interpreters for Python are available on a wide range of platforms like Windows and Linux.

2.9.2 History

The implementation for Python was started by Guido van Rossum in December 1989 in Netherlands. It was designed to succeed ABC and to be capable of exception handling.

Support for Unicode and new features like a garbage were added later in Python 2.0. It was released in October, 2000. The next major iteration was Python 3 which was not backwards compatible. It was released in December 2008. Python 2.6 and 2.7 also borrow many of its features.

CHAPTER:3

IMPLEMENTATION

3.1 PHYSICAL CONNECTIONS

3.1.1 Motor Driver Circuit

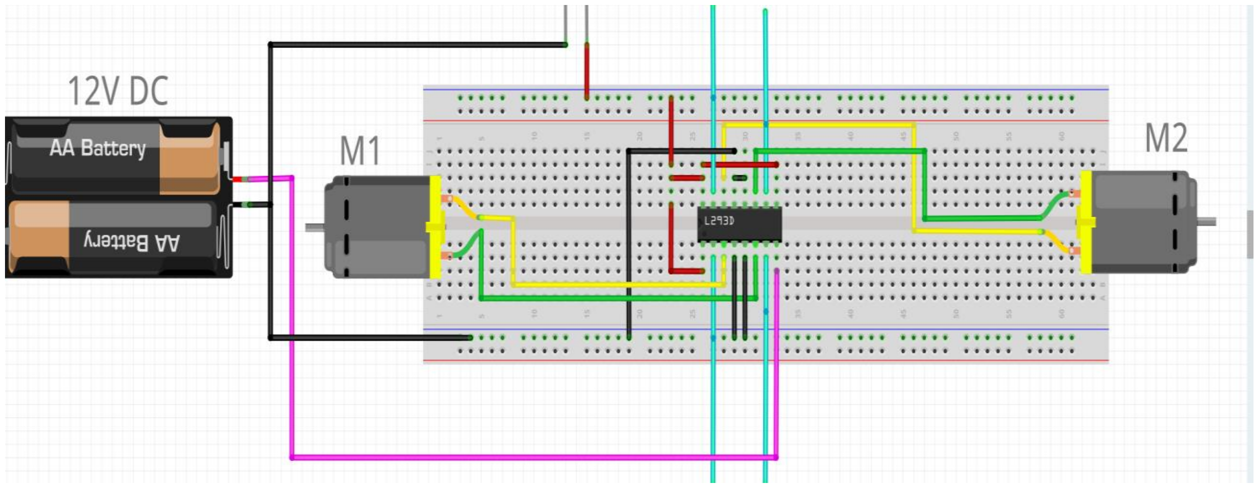

Figure 3.1: Motor driver circuit using L293D

The circuit shown above is the most basic implementation of L293D IC. There are 16 pins sticking out of this IC and we have to understand the functionality of each pin before implementing this in a circuit

1. Pin1 and Pin9 are "Enable" pins. They should be connected to +5V for the drivers to function. If they pulled low (GND), then the outputs will be turned off regardless of the input states, stopping the motors. If you have two spare pins in your microcontroller, connect these pins to the microcontroller, or just connect them to regulated positive 5 Volts.

2. Pin4, Pin5, Pin12 and Pin13 are ground pins which should ideally be connected to microcontroller's ground.
3. Pin2, Pin7, Pin10 and Pin15 are logic input pins. These are control pins which should be connected to microcontroller pins. Pin2 and Pin7 control the first motor (left); Pin10 and Pin15 control the second motor(right).
4. Pin3, Pin6, Pin11, and Pin14 are output pins. Tie Pin3 and Pin6 to the first motor, Pin11 and Pin14 to second motor
5. Pin16 powers the IC and it should be connected to regulated +5Volts
6. Pin8 powers the two motors and should be connected to positive lead of a secondary battery. As per the datasheet, supply voltage can be as high as 36 Volts. [8]

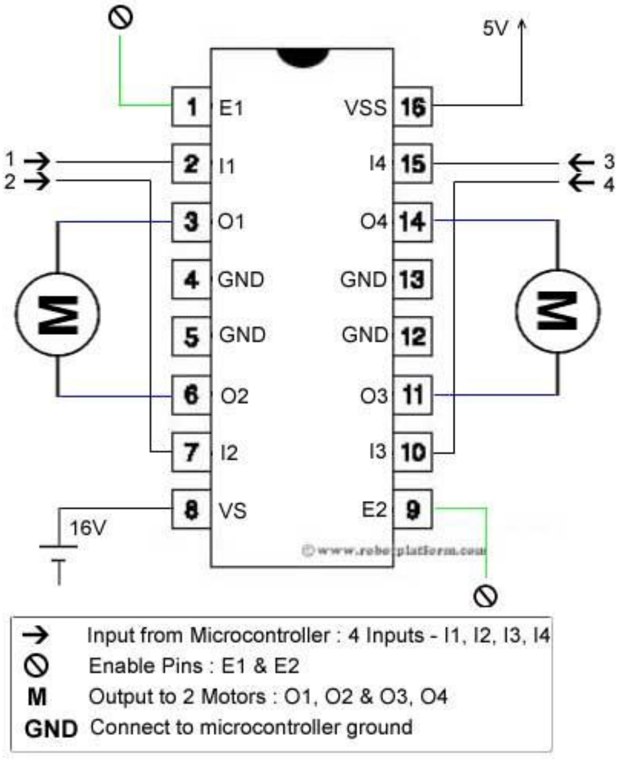

Figure 3.2: Basic implementation of L293D IC

Pin 5v	I1	I2	Function
High	**High**	**Low**	**Turn Anti-clockwise (Reverse)**
High	**Low**	**High**	**Turn clockwise (Forward)**
High	**High**	**High**	**Stop**
High	**Low**	**Low**	**Stop**
Low	**X**	**X**	**Stop**

Table 3.1: logic table for L293D

3.1.2 Raspberry Pi

The Pi is the heart of the project and it is crucial that the connections made to the Pi are correct. The GPIO of the Pi are used to send signals to the motor driver circuit and receive data from the IR sensor. The camera we have used is connected via a USB cable to the port on the board.

On getting the images from the camera, the computer processes the information and accordingly sends out information to the L293D circuit and receives information from the IR sensor once in motion.

The connections made are listed as under (the pin numbering is in GPIO.BOARD convention):

Pin No. on RPi 3	**Connected to**	**Function**
2	Vcc on L293D	Operating power for motor driver circuit
3	Input 3 (Motor 2) on L293D	Input signaling for Motor 2
5	Input 4 (Motor 2) on L293D	
7	Enable 2 (Motor 2) on L293D	Enable signal for Motor 2
8	Input 1 (Motor 1) on L293D	Input signaling for Motor 1
10	Input 1 (Motor 1) on L293D	
12	Enable 1 (Motor 1) on L293D	Enable signal for Motor 1
32	IR sensor output	Receive digital information form IR sensor
34	Ground on IR sensor	Provide power and ground for the IR sensor
36	Vcc on IR sensor	

Table 3.2: Connections for RPi 3

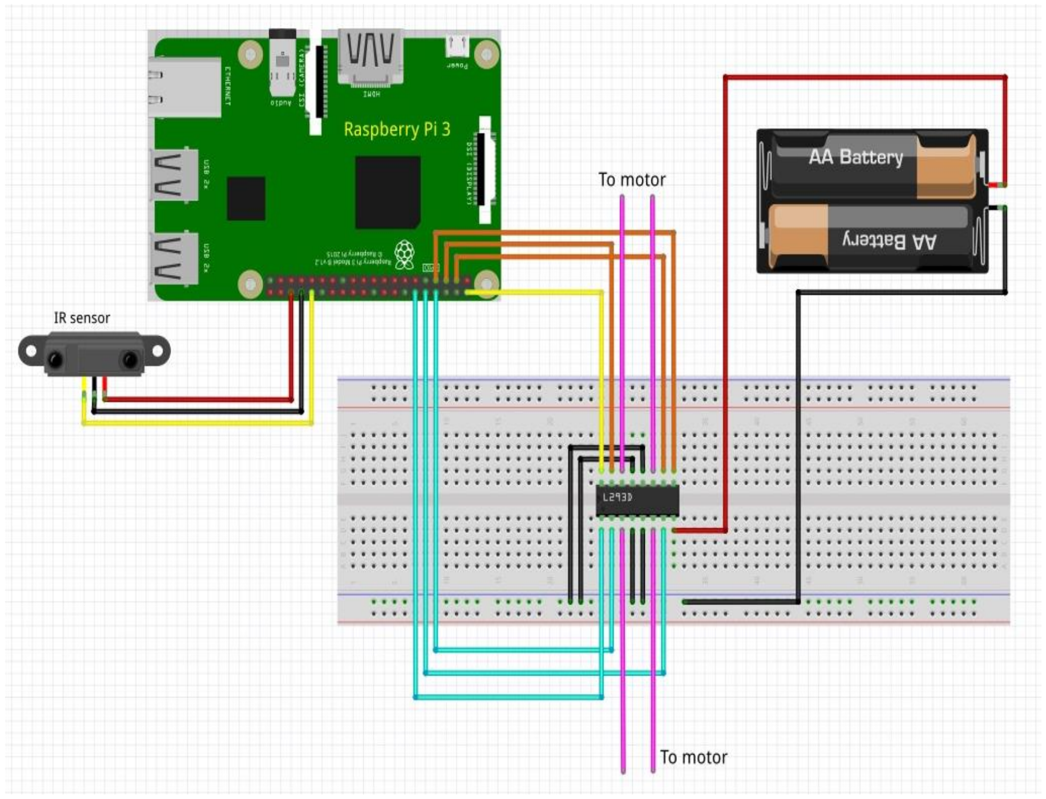

Figure 3.3: Connection between L293d, RPi3and IR Sensor.

3.2 CODE AND ALGORITHM

3.2.1 Installing the OS

The OS we are using for the RPi3 is Raspbian Jessie. [10]For this, we downloaded the image file for the OS from the Raspberry Pi website and using an image writing tool we write the Image onto an SD card. Once the image has been written to the SD card insert it into the Pi and turn on the RPi. Use an Ethernet cable to connect it to the internet or your network, scan your network for the address of the pi and use the address to access the RPi remotely through your laptop via SSH. Once we have access to the RPi, we can do a lot of different things with it. For example, to make working on the project easier we enabled VNC server and use a VNC viewer the remotely access the desktop GUI of the RPi rather than just working in command line. To work on our project, we also installed some more packages which include Wiring Pi, Motion, new JPEG codecs etc.

3.2.2 Installing OpenCV for image processing

Now that we have our OS running and we have remote access to the RPi via SSH, we are ready to install OpenCV to perform our image processing tasks. The list of commands we used in the particular order are[2]:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install build-essential cmake pkg-config
$ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
$ sudo apt-get install libxvidcore-dev libx264-dev
$ sudo apt-get install libgtk2.0-dev
$ sudo apt-get install libatlas-base-dev gfortran
$ sudo apt-get install python2.7-dev python3-dev

$ cd ~
$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.1.0.zip
$ unzip opencv.zip
$ wget -O opencv_contrib.zip https://github.com/Itseez/opencv_contrib/archive/3.1.0.zip
$ unzip opencv_contrib.zip
$ wget https://bootstrap.pypa.io/get-pip.py
$ sudo python get-pip.py
$ sudo pip install virtualenv virtualenvwrapper
$ sudo rm -rf ~/.cache/pip
$ echo -e "\n# virtualenv and virtualenvwrapper" >> ~/.profile
$ echo "export WORKON_HOME=$HOME/.virtualenvs" >> ~/.profile
$ echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile
$ source ~/.profile
$ mkvirtualenv cv -p python2
$ source ~/.profile
$ workon cv
$ pip install numpy
$ cd ~/opencv-3.1.0/
$ mkdir build
$ cd build
$ cmake -D CMAKE_BUILD_TYPE=RELEASE \
   -D CMAKE_INSTALL_PREFIX=/usr/local \
   -D INSTALL_PYTHON_EXAMPLES=ON \
   -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.1.0/modules \
   -D BUILD_EXAMPLES=ON ..
$ make -j4
$ sudo make install
$ sudo ldconfig
$ ls -l /usr/local/lib/python2.7/site-packages/
        total 1852
        -rw-r--r-- 1 root staff 1895772 Mar 20 20:00 cv2.so
```

Checking whether OpenCV was installed correctly or not:

```
$ source ~/.profile
$ workon cv
$ python
>>> import cv2
>>> cv2.__version__
'3.1.0'
>>>
```

The whole process took us about 2.5 hours, of which compiling the OpenCV source code with the "make -j4" command took about half of the time.

3.2.3 Control Flow of Robo-Aid

Robo-Aid is a fully autonomous robot. All it needs is a simple signal to fetch the object needed at the point of origin. On getting the signal to fetch the object, it scans its field of view and checks it for the object by passing the test image clicked through the image detection algorithm (as elaborated in the next section). If the object isn't found, the robot rotates to a new field of view and scans the new area. Once the object is detected, the robot approaches the object and stops on reaching it thanks to the IR sensor at its front. Then the robot returns back to its original position with the object by travelling back the same exact path.

3.2.3.1 Flowchart

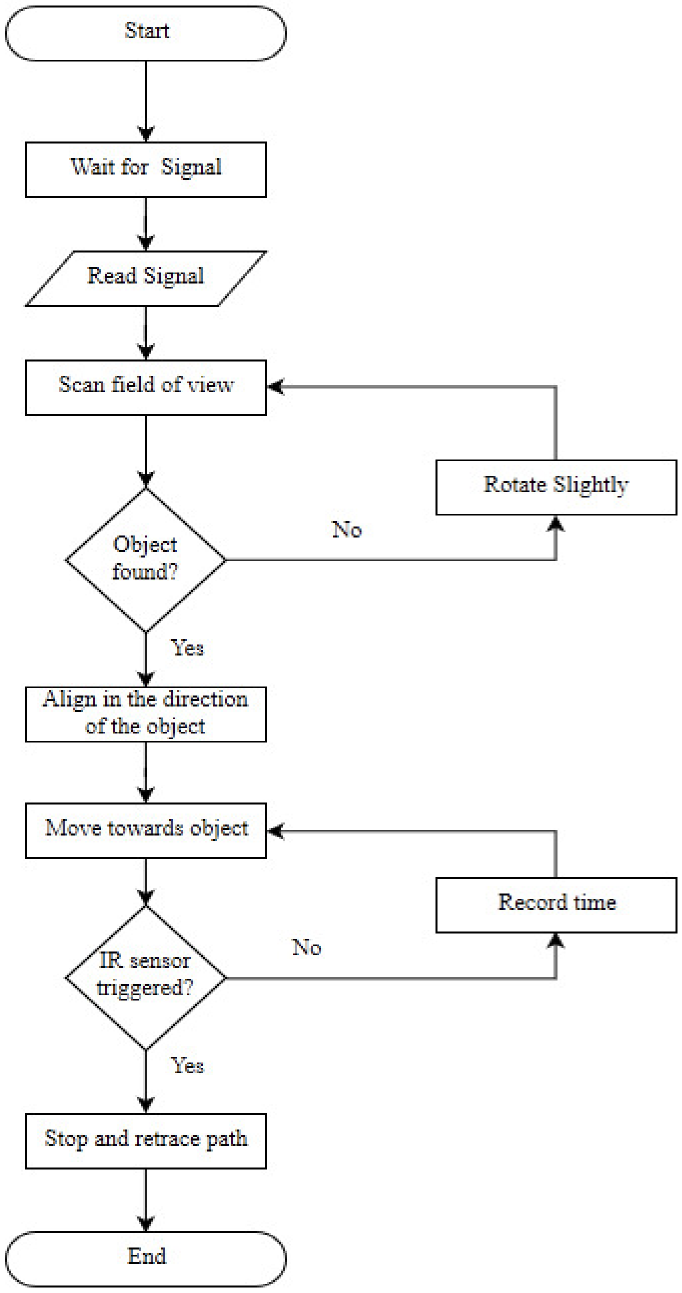

Figure 3.4: Control flow of Robo-Aid

3.2.3.2 Algorithm

1. The robot, once on, waits for signal from the user to retrieve the object.
2. On getting the go from the user, the robot clicks a picture of the field of view.
3. This image is then scanned for the object.
4. If the object is not detected, the robot rotates slightly to check in another direction and thus conducting a 360^o sweep.
5. Once the object is detected, the robot aligns properly in the direction of the object.
6. After aligning, it moves towards the object and turns on the IR sensor when it starts moving.
7. Also, when the robot starts moving, it also starts a timer to calculate the time travelled forward.
8. On reaching the object, the IR sensor is triggered and the robot stops.
9. The robot then retraces its path backward for the same duration with the object to reach its original position.

3.2.3.3 Code

```
import RPi.GPIO as GPIO

import time

import numpy as np

import cv2


#set 38hreshold and load query image

MIN_MATCH_COUNT = 75

img1 = cv2.imread(‘query.jpg’,0)                # queryImage


sift = cv2.xfeatures2d.SURF_create(500)         #Initiate SURF detector

kp1, des1 = sift.detectAndCompute(img1,None)    #KP and DES for query
```

```
bf = cv2.BFMatcher()                                    #initialize matcher


#set GPIO

GPIO.setmode(GPIO.BOARD)

GPIO.cleanup()

GPIO.setup(3,GPIO.OUT)     #IN3

GPIO.setup(5,GPIO.OUT)     #IN4

GPIO.setup(7,GPIO.OUT)     #EN2

GPIO.setup(8,GPIO.OUT)     #IN1

GPIO.setup(10,GPIO.OUT)    #IN2

GPIO.setup(12,GPIO.OUT)    #EN1

GPIO.setup(36,GPIO.OUT)    #IR-EN

GPIO.setup(32,GPIO.IN)     #IR-IN


#initialize

GPIO.output(3,GPIO.LOW)

GPIO.output(5,GPIO.LOW)

GPIO.output(7,GPIO.HIGH)

GPIO.output(8,GPIO.LOW)

GPIO.output(10,GPIO.LOW)

GPIO.output(12,GPIO.HIGH)

while 1:


        #capture image
```

```
cam = cv2.VideoCapture(0)

ret, im=cam.read()

cv2.imwrite(“fintest.jpg”,im)

cam.release()


img2 = cv2.imread(‘fintest.jpg’,0)          # trainImage


# find the keypoints and descriptors with SURF

kp2, des2 = sift.detectAndCompute(img2,None)


#matching

matches = bf.knnMatch(des1,des2,k=2)

# store all the good matches as per Lowe’s ratio test.

Good = []

for m,n in matches:

        if m.distance < 0.8*n.distance:

                good.append([m])




#match conditions

if (len(good)>MIN_MATCH_COUNT):

        break

else:

        GPIO.output(5,GPIO.HIGH)
```

```
                GPIO.output(8,GPIO.HIGH)

                time.sleep(0.1)

                GPIO.output(5,GPIO.LOW)

                GPIO.output(8,GPIO.LOW)


#Move robot to go to the object

GPIO.output(5,GPIO.HIGH)

GPIO.output(10,GPIO.HIGH)


GPIO.output(36,GPIO.HIGH)           #Activate IR sensor

time.sleep(0.1)

n=0

while 1:

        n=n+1

        if(GPIO.input(32) == 1):

                break


GPIO.output(5, GPIO.LOW)

GPIO.output(10, GPIO.LOW)

GPIO.output(36, GPIO.LOW)           #Disable IR sensor

time.sleep(1)


#Reverse the robot to original position

GPIO.output(3, GPIO.HIGH)
GPIO.output(8, GPIO.HIGH)
while n>0:
        GPIO.input(32)
            n=n-1


GPIO.output(3, GPIO.LOW)
GPIO.output(8, GPIO.LOW)


#GPIO clean-up before end
GPIO.cleanup()
```

3.2.4 Object Detection Algorithm

The object detection works in a simple manner. It uses one of the methods for detection of key points and descriptors (available methods in OpenCV library being ORB, SURF, SIFT etc.) for an image to first get a list of key points and their descriptors for both query and test images. The list of descriptors for both images are then passed through a matcher to find out the matching descriptors. The matching descriptors and the corresponding key points are put through a ratio test to check their validity and the valid matches are counted. If the number of valid matches are above a certain predefined thresh hold number, it is a match.

3.2.4.1 Flowchart

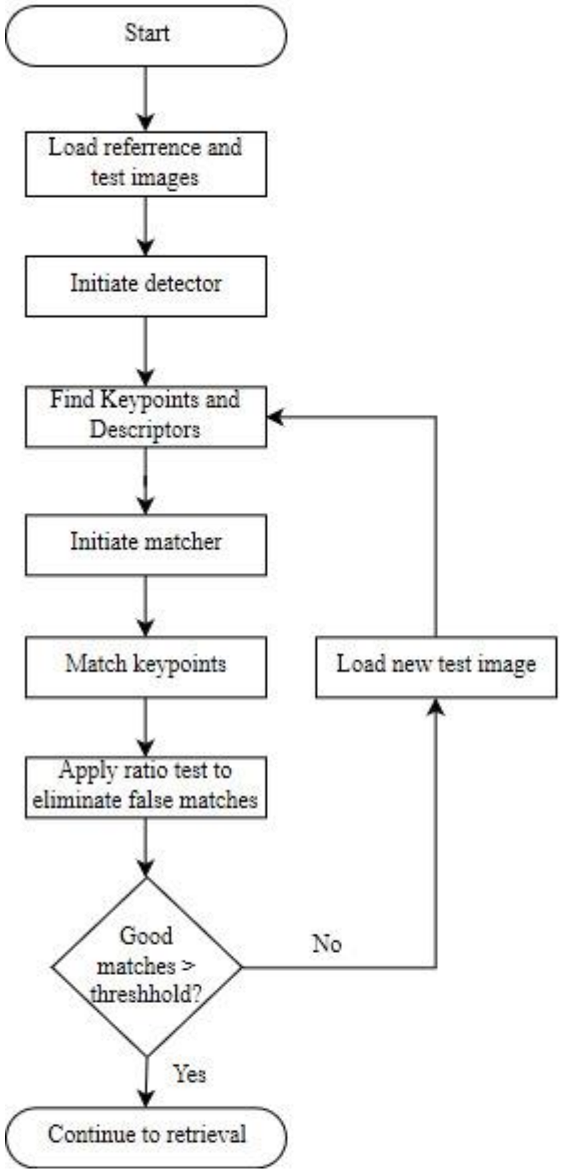
Start
Load referrence and test images
Initiate detector
Find Keypoints and Descriptors
Initiate matcher
Match keypoints
Load new test image
Apply ratio test to eliminate false matches
Good matches > threshhold?
No
Yes
Continue to retrieval

Figure 3.5: Object detection Algorithm

3.2.4.2 Algorithm

1. The query and test images are loaded.
2. Detector is initiated and the key points and their descriptors are calculated for both images.
3. The matcher is initiated and the descriptors (and thus the corresponding key points) are matched.
4. Ratio test is applied to get a list of valid matches.
5. The number of valid matches is compared to the minimum threshold number of matches.
6. If matched the robot goes on to retrieve the object, else the robot rotates to get a new test image to match.

Chapter:4

Results

Over the course of the project we have successfully implemented all the modules and collated them to form our system.
Robo-Aid is working as proposed within the constraints defined in our test environment.

The constraints for optimal functioning of Robo-Aid are:

1. The room or area for functioning should be well lit.
2. The objects to be retrieved are to be kept in standardized boxes which the robot is programmed to recognize.
3. Due to the low resolution of the camera, the object should not be too far.
4. The box to be recognized and retrieved should be in the Field of view.
5. The object will be brought to the point of origin of the bot.

Limitations of the implemented system are as follows:

1. Due to low resolution of the camera, the bot sometimes gives a false positive recognition in poor lighting.
2. Location of objects behind any obstruction or out of field of view cannot be recognized.
3. Objects cannot be too heavy or else the bot might not be able to retrieve the box.

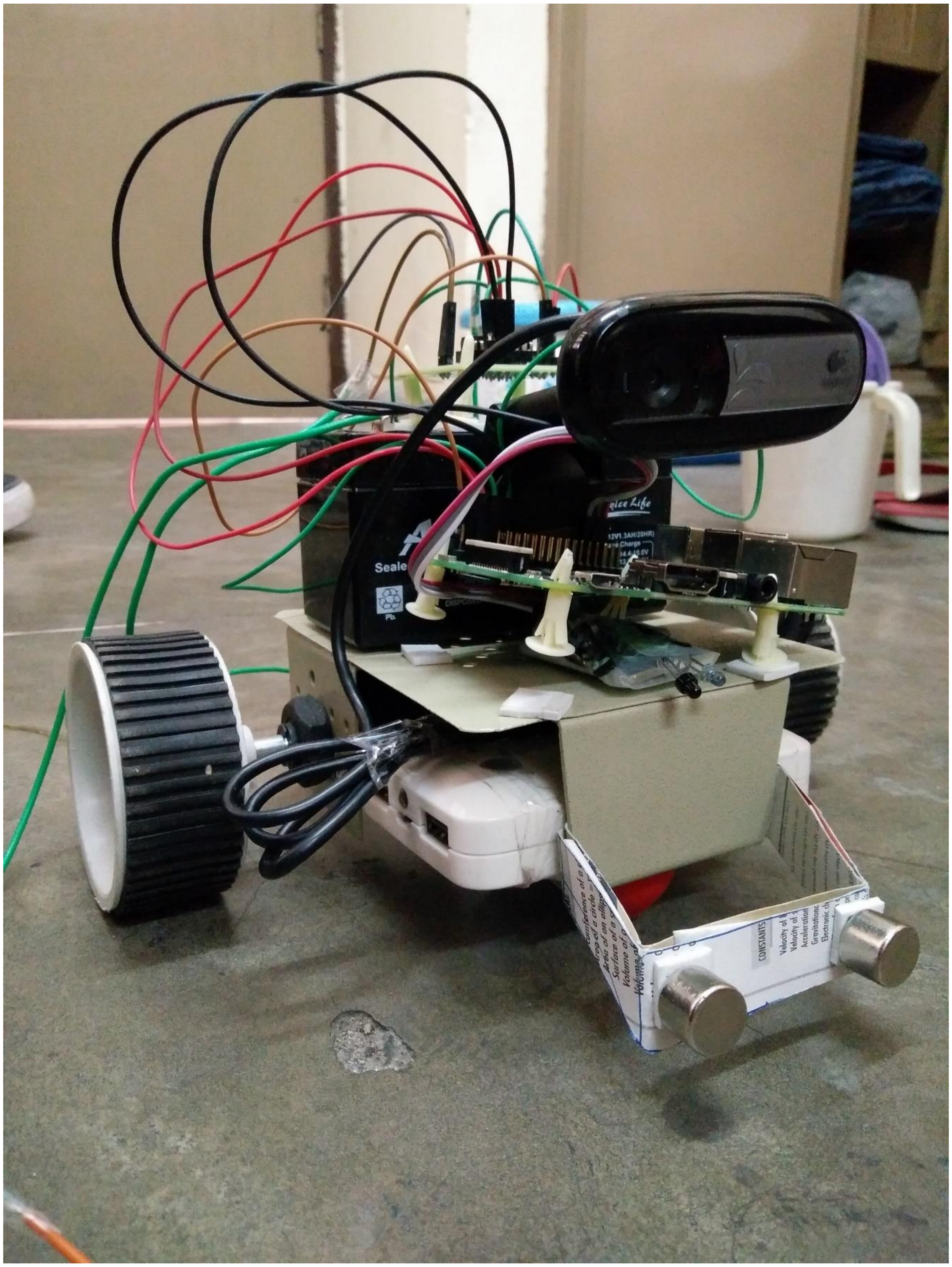

Figure 4.1: Working prototype of the proposed system.

Chapter:5

Future Scope

The system presented here has a lot of room to grow in terms of efficiency, design and adding more features and functions.

The proposed system can be used in many higher-level applications.

For example, auto-nursing bots which give medicine to the sick person at specific times in the day or specific intervals autonomously. Or, a domestic help bot going from room to room fetching objects all over the house and logging their current positions.

The system can also be further developed to be an autonomous child care robot, bringing milk bottle when hearing crying.

References

[1] https://circuitdigest.com/electronic-circuits/ir-sensor-circuit-diagram

[2] http://www.pyimagesearch.com/2016/04/18/install-guide-raspberry-pi-3-raspbian-jessie-opencv-3/

[3] http://www.amptekbatteries.com/smf-industrial-battery.html

[4] https://en.wikipedia.org/wiki/OpenCV

[5] http://www.engineersgarage.com/electronic-components/l293d-motor-driver-ic

[6] http://www.rakeshmondal.info/L293D-Motor-Driver

[7] https://www.element14.com/community/servlet/JiveServlet/previewBody/73950-102-10-339300/pi3_gpio.png

[8] http://www.robotplatform.com/howto/L293/motor_driver_1.html

[9] http://www.robotplatform.com/howto/L293/img_l/L293D_connections.jpg

[10] https://www.raspberrypi.org/documentation/installation/installing-images/

[11] https://www.engineersgarage.com/electronic-components/l293d-motor-driver-ic

[12] https://github.com/opencv/opencv/wiki

[13] https://www.electrical4u.com/working-or-operating-principle-of-dc-motor/

[14] https://circuitdigest.com/electronic-circuits/ir-sensor-circuit-diagram

[15] http://fritzing.org/projects/l293d-dual-h-bridge-motor-driver

[16] http://www.electricaleasy.com/2014/01/basic-working-of-dc-motor.html

[17] http://docs.opencv.org/3.1.0/

[18] http://docs.opencv.org/3.1.0/db/d27/tutorial_py_table_of_contents_feature2d.html