EXPLOSIVE ORDNANCE DISPOSAL ROVER

Thesis submitted in partial fulfillment of the

requirements for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

BY

AKHIL CHERUKURI	**15241A0461**
B JAYANTH REDDY	**15241A0465**
BHARATH BUDDI	**15241A0469**
SUNIL VARMA	**15241A0477**
C SAI MANIKANTA	**15241A0476**
G TUSHAR SAI KUMAR	**15241A0482**

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

GOKARAJU RANGARAJU
INSTITUTE OF ENGINEERING AND TECHNOLOGY
(Approved by AICTE, Autonomous under JNTU Hyderabad)
(Accredited by NBA and NAAC)
Bachupally, Kukatpally, HYDERABAD - 500090
2018-2019

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

GOKARAJU RANGARAJU
INSTITUTE OF ENGINEERING AND TECHNOLOGY
(Approved by AICTE, Autonomous under JNTU Hyderabad)
(Accredited by NBA and NAAC)
Bachupally, Kukatpally, HYDERABAD – 500090

CERTIFICATE

THIS IS TO CERTIFY THAT THE PROJECT REPORT ENTITLED, "**EXPLOSIVE ORDNANCE DISPOSAL ROVER**" IS SUBMITTED BY **AKHIL CHERUKURI (15241A0461)**, **B JAYANTH REDDY (15241A0465), SAI MANIKANTA CHITNENI (15241A0476), G TUSHAR SAI KUMAR (15241A0482), BHARATH BUDDI (15241A0469), AND SUNIL VARMA (15241A0477)** IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING DURING THE ACADEMIC YEAR 2018-2019

(INTERNAL GUIDE)	(HEAD OF DEPARTMENT)
MS. P. LAKSHMI KALA	**DR. N SWETHA**
Assistant Professor, MTech	**Professor, Ph.D.**

(External Examiner)

DECLARATION

I hereby declare that the major project entitled “**EXPLOSIVE ORDNANCE DISPOSAL ROVER**” is the work done during the period from 21 December 2018 to 25 March 2019 and is submitted in the partial fulfillment of the requirements for the award of Degree of Bachelor of Technology in Electronics and Communication Engineering from Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous under Jawaharlal Nehru Technological University, Hyderabad). The results embodied in this project have not been submitted to any other University or Institution for the award of any Degree or Diploma.

Akhil Cherukuri

B Jayanth Reddy

Bharath Buddi

C Sai Manikanta

Sunil Varma

G Tushar Sai Kumar

ACKNOWLEDGMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mentioning of the people whose constant guidance and encouragement made it possible. We take pleasure in presenting before you, our project, which is the result of a studied blend of both research and knowledge.

Firstly, we would like to thank our college administration mainly **DR. J PRAVEEN (Principal)** and **DR. JANDHYALA N MURTHY (Director)** for providing us with world class infrastructure. Then, we would like to thank **DR. N SWETHA (Professor and Head of Department, ECE)** for the undue support and for permitting us to utilize all the necessary facilities of the institution during the course of our project.

We express our earnest gratitude to our internal project guide, **MS. P. LAKSHMI KALA (Assistant Professor, ECE)** for her constant support, encouragement, and guidance. We are grateful for her cooperation and her valuable suggestions. We are also thankful to all the other faculty & staff members of our department for their kind co-operation and help. Finally, we are very indebted to our parents for their moral support and encouragement to achieve higher goals.

Yours sincerely,

Akhil Cherukuri

B Jayanth Reddy

Bharath Buddi

C Sai Manikanta

Sunil Varma

G Tushar Sai Kumar

ABSTRACT

Robots go where humans fear to tread. Of their many applications, bomb disposal is one of the most hazardous, where the risk of death lurks with every move. The main goal of the project is to provide safety to the bomb disposal squad by providing an extra line of defense which is cheap and effective.

System Statement of Scope:

The bomb detectors and disposal system work only with the presence of an expert technician, but this way of analyzing takes more time and make the risk to the life of experts. The Explosive Ordnance Disposal Rover or shortly know as EOD Rover will remove the presence of the bomb detection expert at danger site. The EOD Rover uses a control application, at the user end to control the robot remotely using Wireless technology with Raspberry Pi 3. The bomb technician controls the robot using this application at the control site. Input from the user is transmitted over to the Receiver, where it is received, identified and given to the appropriate commands for the Robot to act. The Robot consists of a Base with tank tires, a robotic Arm and a wireless camera on it. We have used DC motors for the gripper of the robotic arm and at the base for wheel rotation. As we are not risking the life of a bomb expert or technician, hence introducing the safest way for disposing of the explosive to save the life of common people.

Objectives:

1. Provide remote monitoring and controlling application for analysis of a suspicious object.
2. Allow the user to manipulate the packet using the robotic arm.
3. To provide visual feedback from the danger site.
4. To provide a very user-friendly application control interface.

INDEX

PAGE NO.

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

1. RPi	:	Raspberry Pi
2. GPIO	:	General Purpose Input Output
3. CPU	:	Central Processing Unit
4. RAM	:	Random Access Memory
5. ROM	:	Read Only Memory
6. ARM	:	Advanced RISC Machine
7. RISC	:	Reduced Instruction Set Computer
8. BCM	:	Broadcom SOC Channel
9. HTML	:	Hypertext Markup Language
10. CSS	:	Cascading Style Sheets
11. CGI	:	Common Gateway Interface
12. MP	:	Megapixel
13. IED	:	Improvised Explosive Devices
14. PWM	:	Pulse Width Modulation
15. DC	:	Direct Current
16. AI	:	Artificial Intelligence
17. PCB	:	Printed Circuit Board
18. CSI	:	Camera Serial Interface
19. SOC	:	System on Chip
20. USB	:	Universal Serial Bus
21. IDLE	:	Integrated Development Environment
22. MIT	:	Massachusetts Institute of Technology
23. CSI	:	Camera Serial Interface
24. I2C	:	Inter-Integrated Circuit
25. SPI	:	Serial Peripheral Interface
26. RPM	:	Revolutions Per Minute
27. VNC	:	Virtual Network Computing

CHAPTER 1
INTRODUCTION

1.1 GOALS AND OBJECTIVES:

The main goal of the project is to provide safety to the bomb disposal squad by providing an extra line of defense via with the use of remote operated disposal rover.

OBJECTIVES:

- Provide a remote monitoring and controlling application for analysis of a suspicious package or bomb.
- Allow the user to manipulate the packet using the robotic arm.
- To provide visual feedback from the site of the packet.
- To provide a very user-friendly control application.

1.2 SYSTEM STATEMENT OF SCOPE:

The Explosive Ordnance Disposal Rover or shortly know as EOD Rover will remove the presence of the bomb detection expert at danger site. The EOD Rover uses a control application, at the user end to control the robot remotely using Wireless technology with Raspberry Pi 3. The bomb technician controls the robot using this application at the control site. Input from the user is transmitted over to the Receiver, where it is received, identified and given to the appropriate commands for the Robot to act. The Robot consists of a Base with tank tires, a robotic Arm and a wireless camera on it. We have used DC motors for the gripper of the robotic arm and at the base for wheel rotation. As we are not risking the life of a bomb expert or technician, hence introducing the safest way for disposing of the explosive to save the life of common people.

Major Inputs and Outputs:

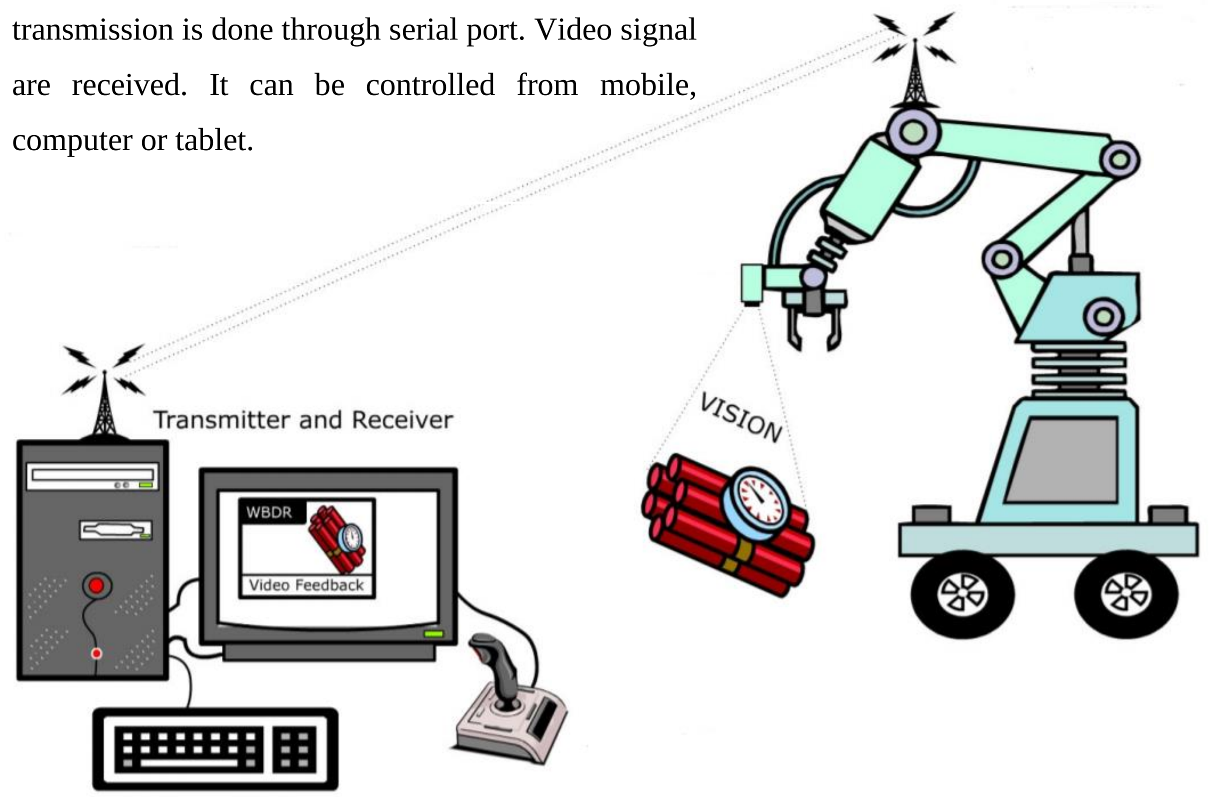
- Input Commands
- Live Video Feedback
- Movement of Base
- Movement of Robotic Gripper

We have designed it as an assistant robot to the bomb disposal squad, but there are some other applications of this robot. It can be used by:

- Police: In hostage situations.
- Military: For reconnaissance missions.
- Fire: To provide video feedback of the site for analysis.
- Nuclear: For handling hazardous or radioactive materials.

1.3 SYSTEM CONTEXT:

Control signals are sent to the robot. Signal transmission is done through serial port. Video signal are received. It can be controlled from mobile, computer or tablet.



Vision signals are transmitted through a transmitter to receiver. Control signals are received with the help of a receiver on the controller attached to the robot which does the complete controlling.

FIGURE 1.1 SYSTEM CONTEXT DIAGRAM

1.4 THEORETICAL BACKGROUND:

The project has been designed keeping in view the current law and order situation in third world countries such as India. If we take a look at the current situation in India administered Jammu and Kashmir, we see hundreds of trained personnel either injured badly or lose their lives while defusing IEDs (Improvised Explosive Devices). This can be reviewed by the countless number of news items appearing daily in newspapers around the world.

Although the idea of our project is original, some projects with similar functionalities can be found. For Example, the British Police have a Bomb Disposal Robot, the Israeli Army has it, and it is also being used by Bomb Disposal Squads near the border between Palestine and some enforcing departments such as FBI, SWAT, ICE in the USA. The main idea of this robot is to provide the bomb disposal squad with safety and security from the risks that they face every day. The bomb disposal squads of J&K have metal detectors and other equipment for bomb detection and disposal, but they have to risk their lives by approaching the bomb or the suspicious packages without any safety and precautions.

Our robot provides an extra layer of protection to the bomb disposal squad by allowing them to check and analyze a suspicious packet before actually approaching it for disposal. Mobile robots reduce or eliminate a bomb technician’s time-on-target. A robot takes the risk out of potentially deadly scenarios and lets the bomb technician focus on what to do to an explosive device rather than on the immediate danger to life and limb. Even if a robot cannot reach an item for disruption, it can still be used to relay information to aid in tool and procedure selection to moving downrange

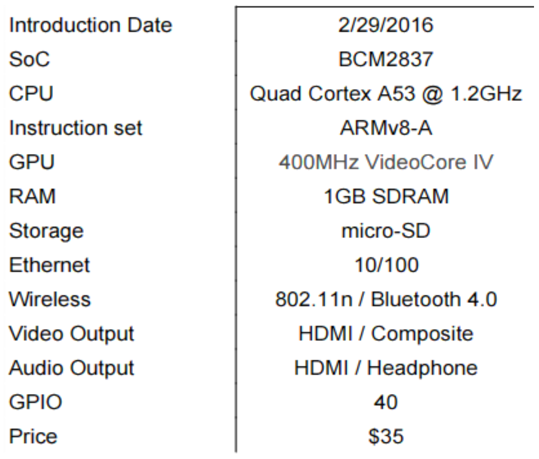
There are two main goals in improving bomb disposal practices: to disarm the device with as little human contact as possible and to save the evidence contained in the bomb. In the past, the first problem has been solved by making robots that can detonate the bomb. This, however, works against the second goal because it destroys the evidence contained within the bomb that can provide law enforcement with the opportunity to find the maker of the bomb. To do this, the bomb must be disarmed without being detonated, a task which is currently almost entirely manual. There is a potential for vast improvement in the ability to accomplish these goals of bomb disposal by using new technology to improve bomb disposal robots and allowing humans to keep a safe distance from the explosives in more situations.

TOOLS AND TECHNOLOGY:

HARDWARE:

1. **Raspberry Pi 3 Model B with 5MP Pi Camera:**

   The Raspberry Pi is a series of small single-board computers developed to promote programming. The Raspberry Pi Camera Board plugs directly into the CSI connector on the Raspberry Pi. It's able to deliver a crystal clear 5MP resolution image at 1080p.



Introduction Date	2/29/2016
SoC	BCM2837
CPU	Quad Cortex A53 @ 1.2GHz
Instruction set	ARMv8-A
GPU	400MHz VideoCore IV
RAM	1GB SDRAM
Storage	micro-SD
Ethernet	10/100
Wireless	802.11n / Bluetooth 4.0
Video Output	HDMI / Composite
Audio Output	HDMI / Headphone
GPIO	40
Price	$35

2. **L298 Motor Driver:**

   The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module will drive DC motors of 12V, with a peak current up to 2A. We will implement the technique of PWM (Pulse Width Modulation) to control the motors.

3. **SG90 Servo Motor:**

   It is tiny and lightweight with high output power. This servo can rotate approximately 180 degrees (90 in each direction). These Motors are used for the movement of the Robotic Arm. PWM signal is given in through the transmission wire to drive the motor.

4. **12V Motors:**

   Heavy duty DC Motors are used in the chassis for the movement of the device. The DC motors are rated at 12V with 150W at 13000 RPM. This 775 Motor Micro DC Motor also has a 5mm Shaft Motor. Here two DC motors are used, one for right track and one for left track.

SOFTWARE:

1. **EAGLE PCB DESIGN SOFTWARE:**
   EAGLE by Autodesk has been used in PCB designing in order to make the system has clean as possible without excess wirings.

2. **RASPBIAN SOFTWARE:**
   RASPBIAN OS is a Debian-based computer operating system for Raspberry Pi.

3. **PYTHON LANGUAGE:**
   Python is a powerful multi-purpose programming language. It has simple easy-to-use syntax, making it the perfect language for Raspberry Pi. This is the base of the project and will be running in the background.

4. **HTML LANGUAGE:**
   HTML is the standard markup language for creating Web pages and is used in this project for Web Interface between controller and receiver.

5. **CSS LANGUAGE:**
   CSS is the language for describing the presentation of Web pages and is used to make the web interface look exemplary.

6. **COMMON GATEWAY INTERFACE:**
   It is a specification for transferring information between a World Wide Web server and a CGI program. A CGI program is any program designed to accept and return data that conforms to the CGI specification.

7. **MIT APP INVENTOR 2:**
   App Inventor 2 is an open-source cloud-based tool maintained by the Massachusetts Institute of Technology and is used to make Android Applications.

8. **JAVASCRIPT LANGUAGE:**
   Web Interface Language, it is used in this project to display a live stream of the video.

CHAPTER 2
LITERATURE SURVEY

Title 1: Internet of Things with Raspberry Pi 3
Authors: Maneesh Rao
Publisher: Packt Publishing Limited

Abstract: This book is designed to introduce you to IoT and Raspberry Pi 3. It will help you create interesting projects such as a real-world project by building a Wi-Fi - controlled robot car with Raspberry Pi using a motor driver circuit, DC motor, and a web application. This book provides a practical overview of IoT's existing architectures, communication protocols, and security threats at the software and hardware levels-security being the most important aspect of IoT. You will learn to understand the concept of IoT and get familiar with the features of Raspberry Pi Learn to integrate sensors and actuators with the Raspberry Pi. It also helps us understand how to communicate with cloud and using communication protocols such as HTTP, build Do It Yourself projects using Raspberry Pi and JavaScript.

Title 2: Programming the Raspberry Pi: Getting Started with Python (2nd Edition)
Authors: Simon Monk
Publisher: McGraw-Hill Education

Abstract: This updated guide to programming your own Raspberry Pi projects. The books help us learn to create inventive programs and fun games on your powerful Raspberry Pi 3 with no programming experience required. This practical book has been revised to fully cover the new Raspberry Pi 3, including upgrades to the Raspbian operating system. Discover how to configure hardware and software, write Python scripts, create user-friendly GUIs, and control external electronics. Helps us to write Python programs using the IDLE editor. Lessons on how to use strings, lists, functions, and dictionaries. Also, it has work with modules, classes, and methods and shows how to attach external electronics through the GPIO port and helps us to add powerful Web features to your projects.

Title 3: Learn Robotics with Raspberry Pi: Build and Code Your Own Moving, Sensing, Thinking Robots
Authors: Matt Timmons-Brown
Publishers: No Starch Press

Abstract: Learn Robotics with Raspberry Pi will take you from inexperienced maker to robot builder. You'll start off building a two-wheeled robot powered by a Raspberry Pi minicomputer and then program it using Python, the world's most popular programming language. Gradually, you'll improve your robot by adding increasingly advanced functionality until it can follow lines, avoid obstacles, and even recognize objects of a certain size and color using computer vision.

Title 4: Learning Python with Raspberry Pi
Authors: Alex Bradbury and Ben Everard
Publishers: John Wiley & Sons

Abstract: Raspberry Pi chose Python as its teaching language of choice to encourage a new generation of programmers to learn how to program. This approachable book serves as an ideal resource for anyone wanting to use Raspberry Pi to learn to program and helps you get started with the Python programming language. Python is a simple yet powerful programming language that can enable you to start thinking like a programmer right from the beginning. It is very readable and the stress many beginners face about memorizing arcane syntax typically presented by other programming languages will not affect you at all. Conversely, you will be able to concentrate on learning concepts and paradigms of programming. By reading the book and implementing what you learn herein, you will realize just why major institutions prefer to use python in their core products, services and business processes. Aimed at first-time developers with no prior programming language.

Title 5: Arduino Based Battlefield Assistive Robot
Authors: Ahsanul Hoque, Md. Baijid Hasan Shorif, Md. Eftekhar Alam
Publishers: IEEE (R10-HTC)

Abstract: Military forces now using robots for reducing causalities and to defeat their enemies. The major focus of this project, is on the use of robot in war, peace and as well as their impact on society. Here Radio Frequency modules signals are used in wireless remote-control system for transmitting and receiving wireless signals to control the motors and actuators of robot control system. Night vision monitoring system has been added which will capture and transmit the information surrounding the robot to the operator, with this feature the robot can transmit real time videos with night vision. A metal detector and GSM module has also been added which will inform us about any bomb underneath the robot vehicle. Another assistive feature here added that, is a robotic arm has been installed to pick or drop some object if needed.

Title 6: Application of Radio Frequency Controlled Intelligent Military Robot in Defense

Authors: Saradindu Naskar, Asoke Nath, Abhik Kumar Seth

Publishers: IEEE (2010-CSNT)

Abstract: In the present paper the authors tried to explore how a radio frequency-controlled robot can be used in defense and in real war field. The military robot will be able to substitute the real human soldier in the battle field. The authors have tried to explore how a military robot will function. The military robot has a two-barrel gun turret through which bullets can be fired. It has two cameras in synchronization with the turret which can rotate up and down, left and right up to a safe firing limit. The robot vehicle can move like a tank, turning to any angle on its axis, moving forward and reverse turning left and right, running instantly into reverse direction using the same steering mechanism as present in tanks. The robot is radio operated, self-powered, has back tracking facility, in case of loss of connection from the base station. Wireless cameras will send back real time video and audio inputs which can be seen on a remote monitor in the base station from where the robot is being controlled and action can be taken accordingly. The robot can be controlled from a base station by means of radio frequency. It also has the ability to re-establish contact with the base station in case of a signal failure by retracing its path back for some distance. It can silently enter into an enemy area and send us all the information through its camera eyes. It is designed for fighting, reconnaissance as well as suicide attacks under certain circumstances.

Title 7: Multipurpose Military Service Robot

Author: E Amareswar, K R Maheshwari, E Akhil, T Naveen

Publication: IEEE (ICECA 2017)

Abstract: Robotics has been a staple of advanced manufacturing for over half a century. As robots and their peripheral equipment become more sophisticated, reliable and miniaturized, these systems are increasingly being utilized for military and law enforcement purposes Mobile robotics play an increasingly important role in military matters, from patrol to dealing with potential explosives. There are multiple applications on the internet that exploit inbuilt hardware in these mobile phones, such as Wi Fi technology to control other devices. With the development of modern technology and Android Smartphone, technology aims to exchange data wirelessly at a short distance using radio wave transmission comprising features to create ease, perception and controllability. In this paper we have designed a robot that can be controlled using an application running on an android phone. According to commands received from android the robot motion can be controlled.

Title 8: Novel EOD Robot Design with Dexterous Gripper
Author: Matthew W. Carey, Eric M. Kurz, Joshua D. Matte
Publication: WORLD AUTOMATION CONGRESS, 2012

Abstract: It details the design and implementation of an intelligent explosive ordnance disposal robot to provide law enforcement agencies with a cost effective and reliable robotic platform. The key features of the robot include an intuitive user interface which provides additional sensor feedback and enhanced visual awareness compared to existing systems, an on-board three degree of freedom manipulator arm providing an enlarged workspace, and a dexterous gripper allowing for the removal of blasting caps. The flexible and modular robot design utilizes commercial off-the-shelf components for ease of maintenance and repairs. The robot provides a safe distance threat assessment and increased capacity for explosive ordnance disposal, improving the effectiveness of bomb disposal teams. The robot's low-cost, intuitive operation and ease-of-maintenance promote its widespread appeal, thereby saving the lives of both law enforcement personnel and civilians.

CHAPTER 3
FUNCTIONAL AND DATA DESCRIPTION

3.1 SYSTEM ARCHETECURE:

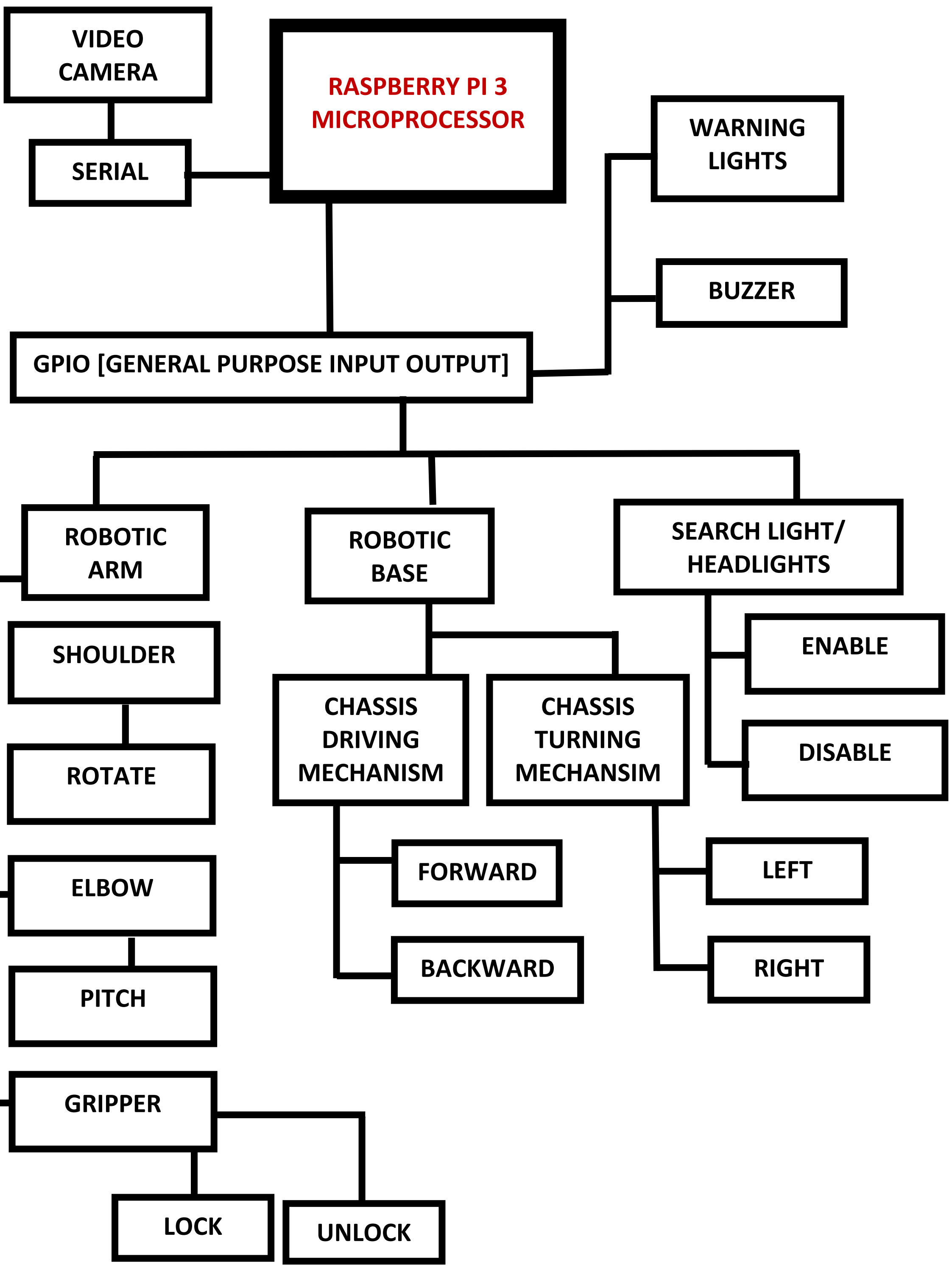


FIGURE 3.1 SYSTEM ARCHITECTURE

3.2 ARCHITECTURE MODEL:

The architecture of Ordnance Disposal Rover is based on a modular approach. For every function there is a separate module, which performs its individual action. The architecture of the rover consists of the following modules:

SOFTWARE ARCHITECTURE:

The application is divided into the following modules according to the functions that they perform:

VIDEO INTERFACE MODULE:

This module is used to initialize the video card and also to control the live feedback of the video, which is transmitted from the robot to the control application. This module also contains the facility to record the video and store it in the computers using third party applications.

VIRTUAL KEYBOARD/ TOUCH INTERFACE MODULE:

This module is used for the control and actions for the robotic arm, which is controlled by the buttons and the changes occurring with the touch of buttons results in the different movements of the robotic arm. We will have the ability to control the modules using keyboard inputs or via touch inputs.

SERIAL INTERFACE MODULE:

Serial communication is accomplished using this module. This module monitors all the data sent through the serial port and all the synchronization of the microcontroller and this module does serial port.

APPLICATION CONTROL MODULE:

All the events occurring when using the software application are controlled through this module. This means all the application level design and inputs of the end user are translated into control signals, which are transmitted to the microcontroller. All the user interface and error control are done in this module.

3.3 ROBOTIC ARCHITECTURE:

The application is divided into the following modules according to the functions that they perform:

ROBOTIC BASE:

The base is made of rectangular aluminum sheet with two motors attached to the back two wheels on either side. The wheels are attached to a rubber track for easy movability. The base consists of the following mechanisms that are responsible for driving and turning.

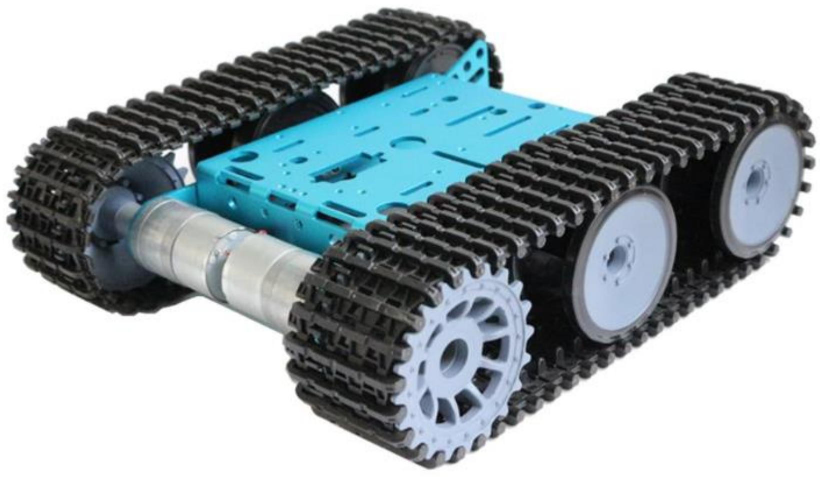


FIGURE 3.2 TANK CHASSIS

DC MOTORS:

The Driving DC motors are 12-volt DC motors that are connected to each side-tracked wheel of the robot base. Torque from the DC motor is transferred from tracked wheel to the individual wheels of the particular side to drive the robot.

ROBOTIC ARM:

The robotic arm has been designed using acrylic strips. The power to weight ratio has been kept in mind in the design of the robotic arm, as it has to be light enough to be moved by the SG90 Plastic Gear Servo and strong enough to be able to move nominal weight objects.

It is a jointed arm type robotic arm with 3 degrees of freedom. It consists of:

1. A Stepper motor for the shoulder of the robotic arm. This stepper motor is mounted on a circular plate onto which two acrylic strips have been mounted.

2. At the next joint, the elbow, a Servo motor is fixed which is responsible for the pitch of the robotic arm. This Servo motor is responsible for the vertical movement of the arm.

3. The final joint is connected to a Servo, which controls the movement of the gripper. The gripper is a two-finger type and can pick up objects with ease.

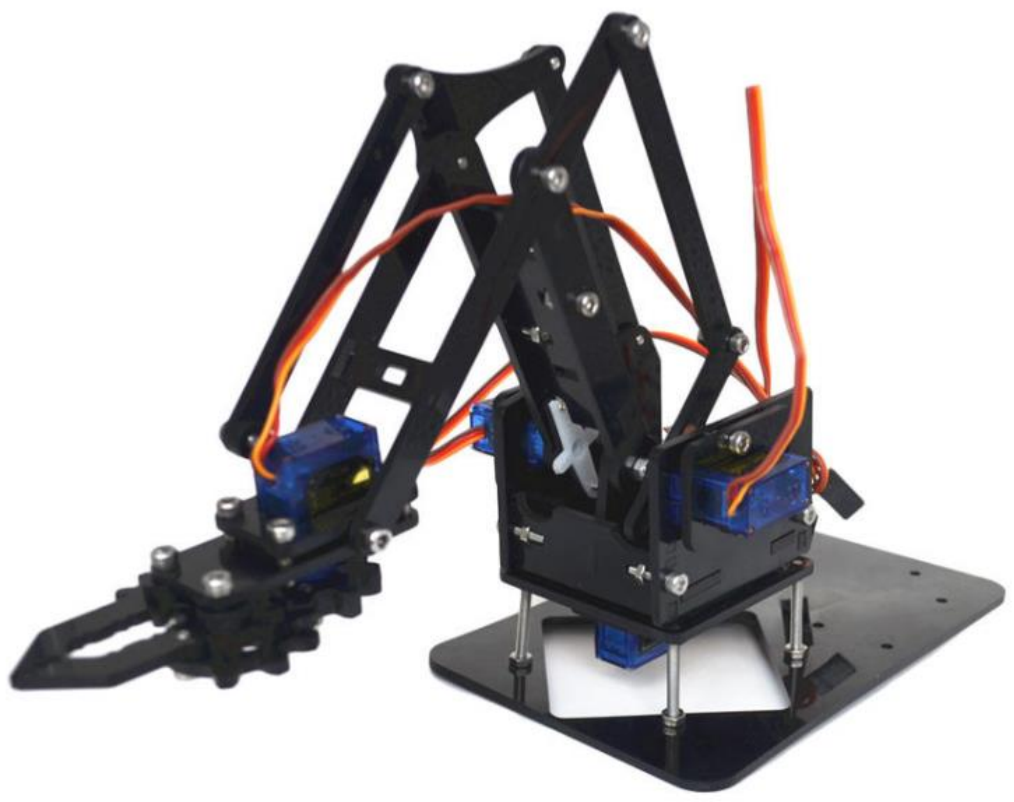


FIGURE 3.3 SERVO MOTOR POWERED ROBOTIC ARM

CHAPTER 4
SUBSYSTEMS

4.1 DESCRIPTION OF SUBSYSTEMS:

4.1.1 SUBSYSTEM SCOPE:

Each subsystem has been designed using a modular approach. The subsystems can be connected and disconnected from other subsystems very easily. A modular approach on the subsystems makes it easy to trouble shoot them, consuming less time.

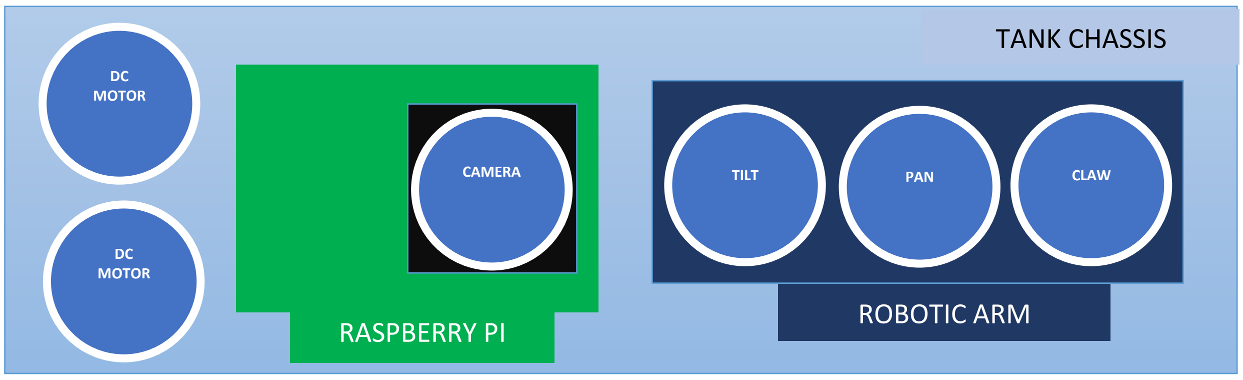


FIGURE 4.1 VARIOUS SUBSYSTEMS

4.2 DESCRIPTION FOR COMPONENT:

The base of the robot consists of the following subsystems:

4.2.1 RASPBERRY PI:

This is the brain of the robot and is known as Single-board computer. Its purpose is to generate the control signals required to control different modules of the robot. The Raspberry Pi includes, the Broadcom BCM2837 system-on-chip (SoC) includes four high-performance ARM Cortex-A53 processing cores running at 1.2GHz, a VideoCore IV graphics processor, and is linked to a 1GB LPDDR2 memory module on the rear of the board. Broadcom BCM43438 chip provides 2.4GHz 802.11n wireless LAN and Bluetooth 4.1. It features the 40-pin general-purpose input-output (GPIO) header to which external modules can be connected and controlled.

4.2.1.1 GPIO:

A powerful feature of the Raspberry Pi is the row of GPIO (general-purpose input/output) pins along the top edge of the board. A 40-pin GPIO header is found on all current Raspberry Pi boards. Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes.

VOLTAGES: Two 5V pins and two 3V3 pins are present on the board, as well as a number of ground pins (0V), which are unconfigurable. The remaining pins are all general purpose 3V3 pins, meaning outputs are set to 3V3 and inputs are 3V3-tolerant.

OUTPUTS: A GPIO pin designated as an output pin can be set to high (3V3) or low (0V)

INPUTS: A GPIO pin designated as an input pin can be read as high (3V3) or low (0V). This is made easier with the use of internal pull-up or pull-down resistors. Pins GPIO2 and GPIO3 have fixed pull-up resistors, but for other pins this can be configured in software.

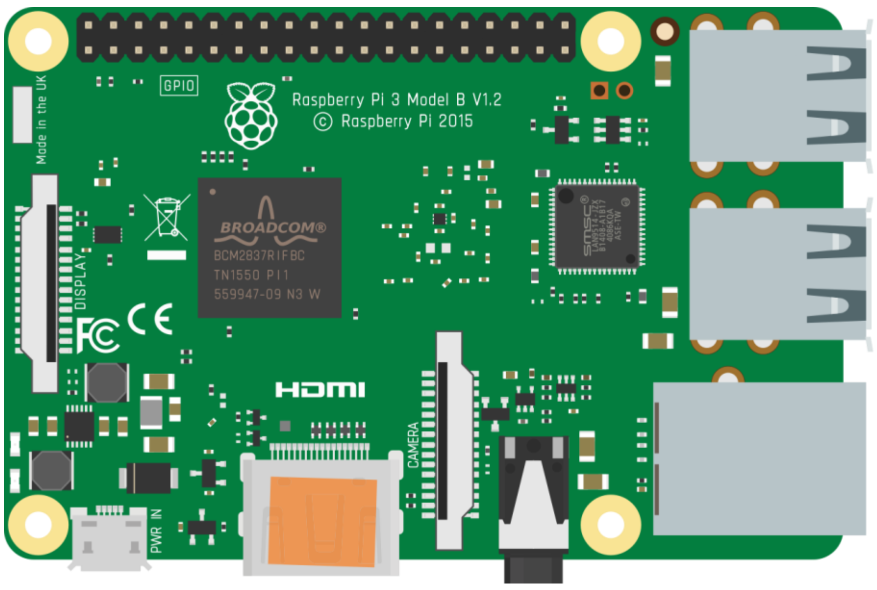


FIGURE 4.2 RASPBERRY PI 3 MODEL B

OTHERS: As well as simple input and output devices, the GPIO pins can be used with a variety of alternative functions, some are available on all pins, others on specific pins.

1. PWM (pulse-width modulation):
   Software PWM available on all pins
   Hardware PWM available on GPIO12, GPIO13, GPIO18, GPIO19

2. SPI:

   SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1 (GPIO7)

   SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18); CE1 (GPIO17); CE2 (GPIO16)

3. I2C:

   Data: (GPIO2); Clock (GPIO3)

   EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)

4. Serial:

   TX (GPIO14); RX (GPIO15)

GPIO PINOUT:

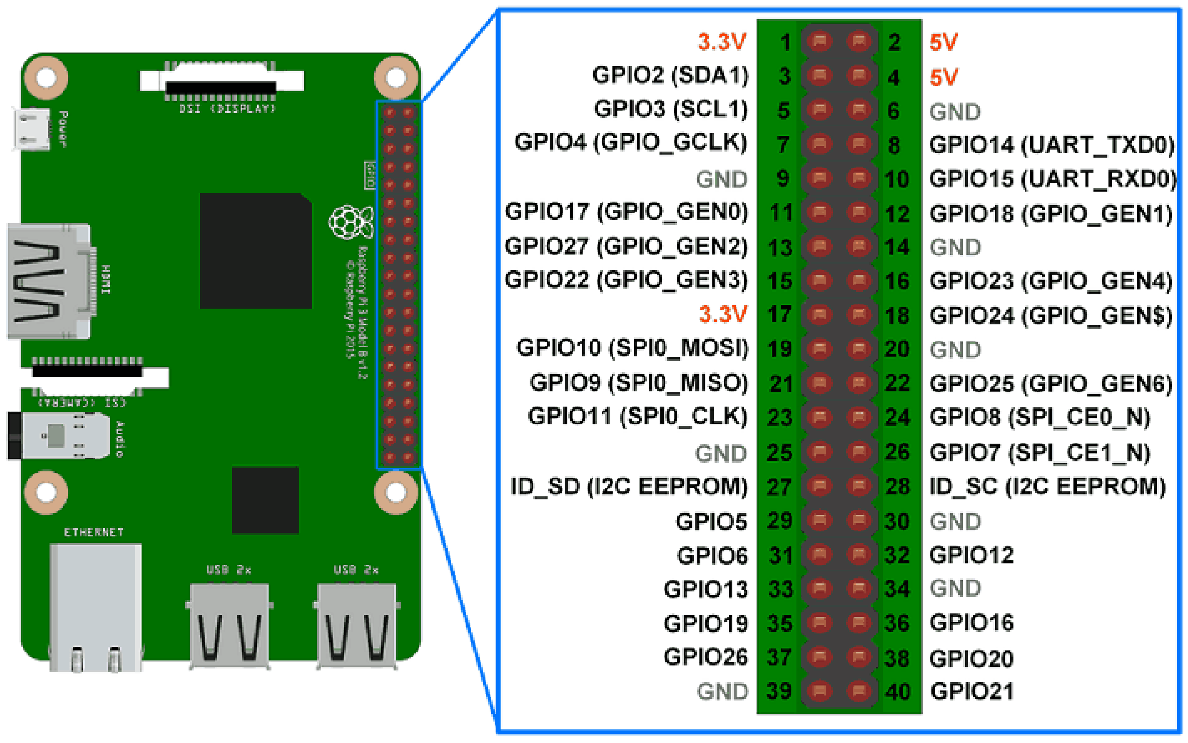

DSI (DISPLAY)
Power
GPIO
HDMI
Raspberry Pi 3 Model B v1.2
© Raspberry Pi 2015
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
3.3V 1 2 5V
GPIO2 (SDA1) 3 4 5V
GPIO3 (SCL1) 5 6 GND
GPIO4 (GPIO_GCLK) 7 8 GPIO14 (UART_TXD0)
GND 9 10 GPIO15 (UART_RXD0)
GPIO17 (GPIO_GEN0) 11 12 GPIO18 (GPIO_GEN1)
GPIO27 (GPIO_GEN2) 13 14 GND
GPIO22 (GPIO_GEN3) 15 16 GPIO23 (GPIO_GEN4)
3.3V 17 18 GPIO24 (GPIO_GEN$)
GPIO10 (SPI0_MOSI) 19 20 GND
GPIO9 (SPI0_MISO) 21 22 GPIO25 (GPIO_GEN6)
GPIO11 (SPI0_CLK) 23 24 GPIO8 (SPI_CE0_N)
GND 25 26 GPIO7 (SPI_CE1_N)
ID_SD (I2C EEPROM) 27 28 ID_SC (I2C EEPROM)
GPIO5 29 30 GND
GPIO6 31 32 GPIO12
GPIO13 33 34 GND
GPIO19 35 36 GPIO16
GPIO26 37 38 GPIO20
GND 39 40 GPIO21

FIGURE 4.3 GPIO PINOUT

4.2.2 L298 MOTOR SHIELD:

The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A.

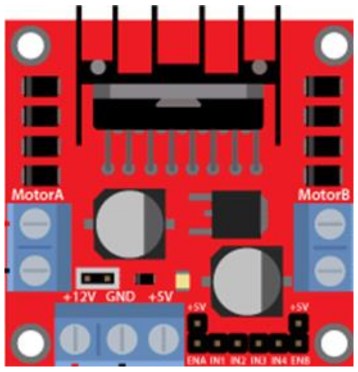
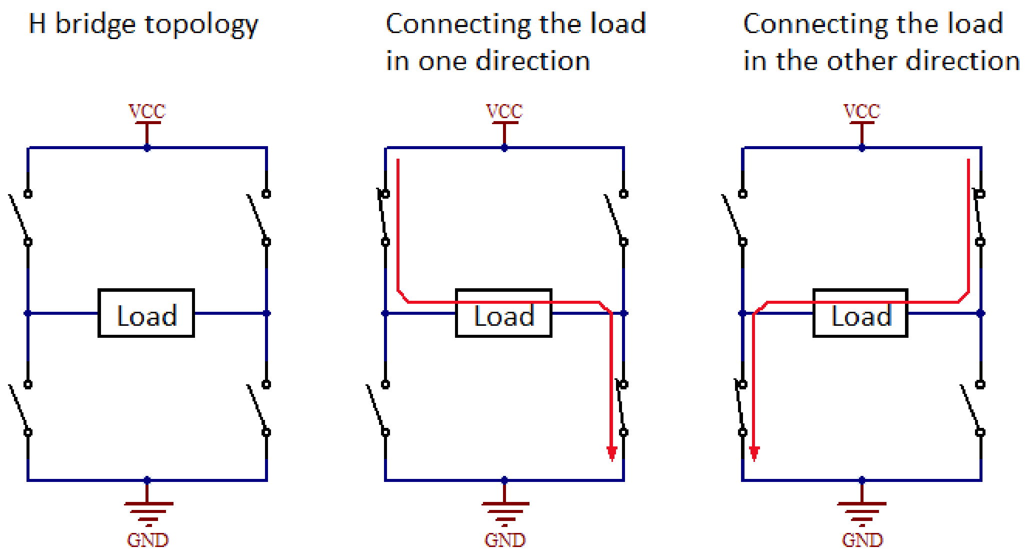


FIGURE 4.4 L298N MOTOR SHEILD

For controlling the rotation direction, we just need to inverse the direction of the current flow through the motor, and the most common method of doing that is by using an H-Bridge. An H-Bridge circuit contains four switching elements, transistors or MOSFETs, with the motor at the center forming an H-like configuration. By activating two particular switches at the same time we can change the direction of the current flow, thus change the rotation direction of the motor.



H bridge topology
Connecting the load in one direction
Connecting the load in the other direction
VCC
Load
GND

FIGURE 4.5 H BRIDGE TOPOLOGY

Here,

Pin 12V = Connected to 12V power supply, this voltage is consumed by the two-heavy duty 12V motors which are mounted on the chassis.

Pin 5V = Connected to 5V power supply, this is used to power up the motor driver in L298

Pin GND = All the pins given in shield are grounded by GND pin

Pin EA1 and EA2 = Control Motor A:

EA1 HIGH, EA2 LOW = Clockwise direction

EA2 HIGH, EA1 LOW = Anticlockwise direction

EA1, EA2 LOW/HIGH = No direction

Pin EA3 and EA4 = Control Motor B:

EA3 HIGH, EA4 LOW = Clockwise direction

EA4 HIGH, EA3 LOW = Anticlockwise direction

EA3, EA4 LOW/HIGH = No direction

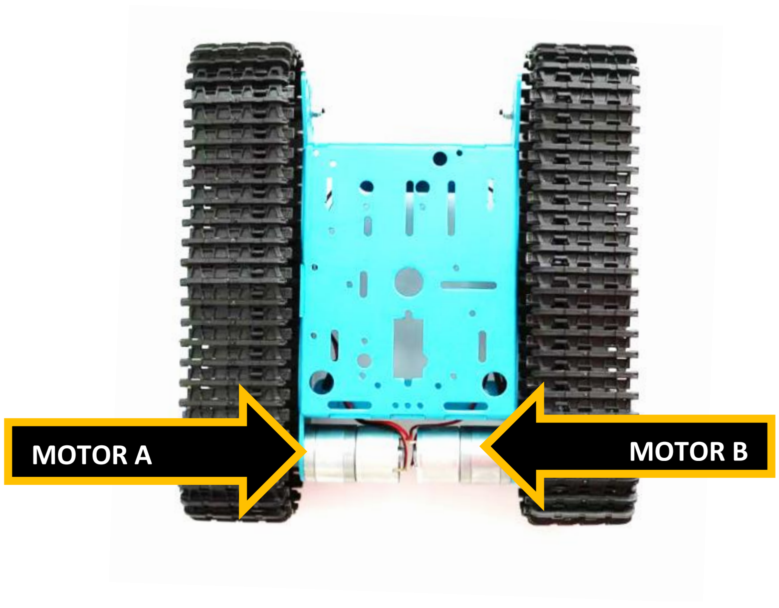


FIGURE 4.6 MOTOR A AND B

We are using a tank chassis to enable movement of rover. Tanks turn by varying the speed of the tracks on one side, causing the tank to turn in the direction of the slower or stopped track or by turning the tracks in opposite directions. One track goes forward, the other in reverse. This can allow the tank to turn in place.

The directions of the two motors installed on tank are expressed as:

DIRECTION	EA1 (Motor1)	EA2 (Motor1)	EA3 (Motor2)	EA4 (Motor2)
RIGHT	HIGH	LOW	LOW	HIGE
LEFT	LOW	HIGH	HIGH	LOW
STOP	LOW	LOW	LOW	LOW
FRONT	LOW	HIGH	LOW	HIGH
REVERSE	LOW	HIGH	LOW	HIGH

TABLE 4.1 MOTOR DIRECTION

4.2.3 ROBOTIC ARM CONTROL CIRCUIT:

Robotic Arm is controlled by using SG90 Servo Motors for the arms various movements. All motors have three wires coming out of them. Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU.

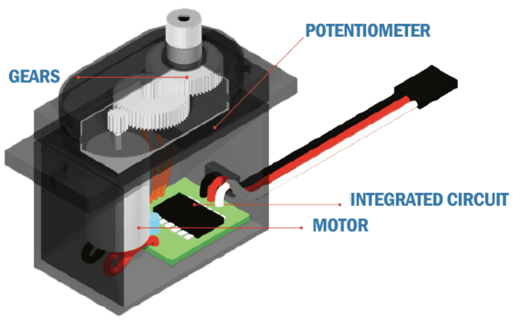


FIGURE 4.7 INSIDE SG90 SERVO MOTOR

Servo motor works on PWM (Pulse width modulation) principle, means its angle of rotation is controlled by the duration of applied pulse to its Control PIN. Basically, servo motor is made up of DC motor which is controlled by a variable resistor (potentiometer) and some gears. High speed force of DC motor is converted into torque by Gears. In Servo, force is High and distance is less. Potentiometer is connected to the output shaft of the Servo, to calculate the angle and stop the DC motor on required angle

Servo motor can be rotated from 0 to 180 degree. This degree of rotation can be controlled by applying the Electrical Pulse of proper width, to its Control pin. Servo checks the pulse in every 20 milliseconds. Pulse 1 millisecond width can rotate servo to 0 degree, 1.5ms can rotate to 90 degree (neutral position) and 2 milliseconds pulse can rotate it to 180 degree.

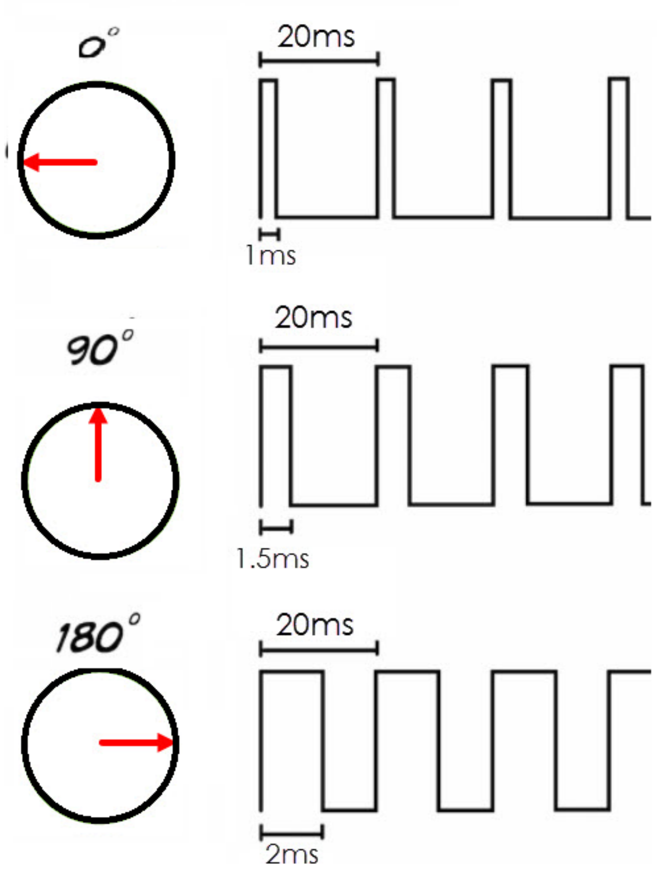


FIGURE 4.8 PWM PRINCIPLE FOR DIRECTION

4.2.4 POWER SUPPLY CIRCUIT:

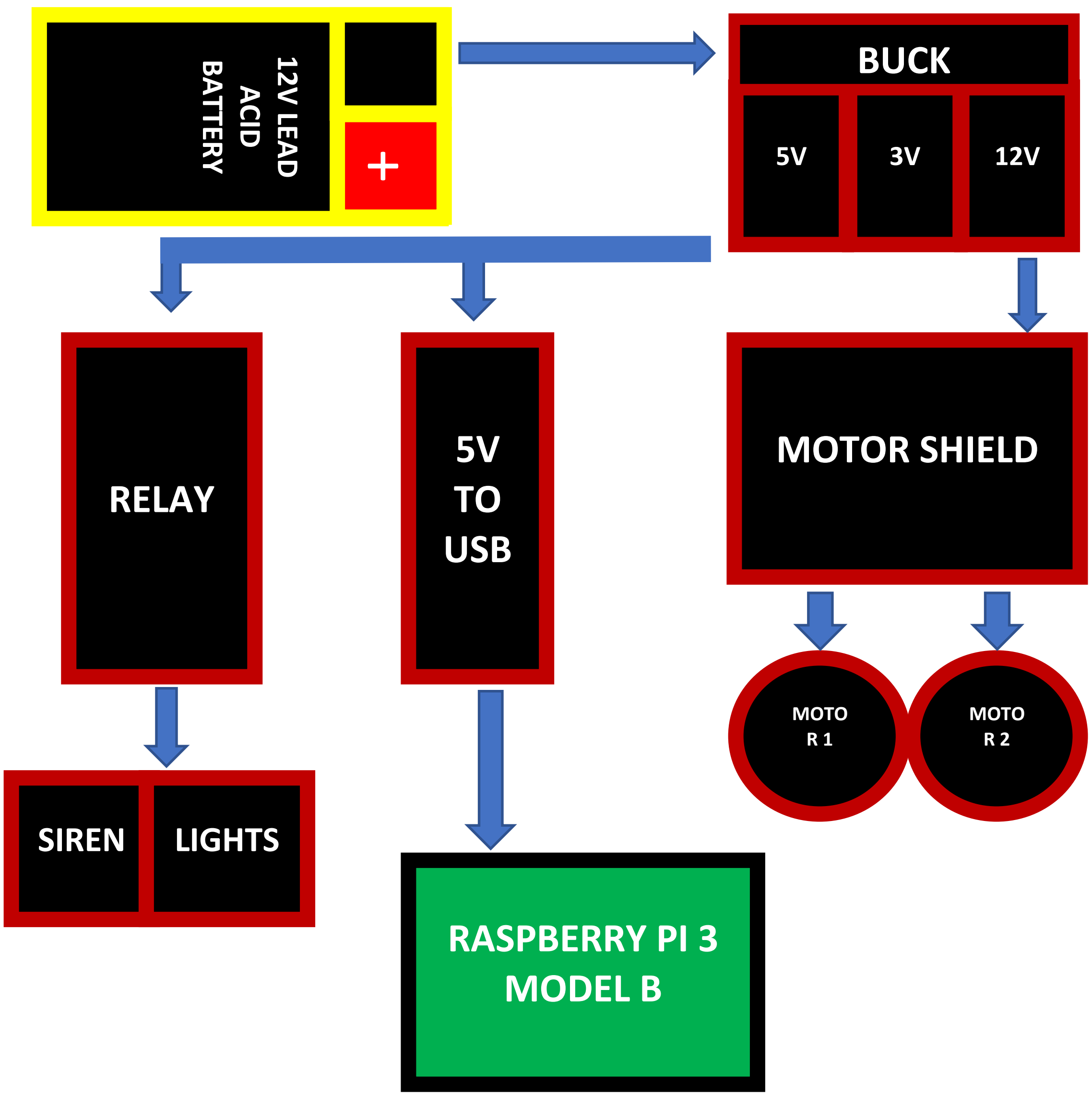


FIGURE 4.9 POWER SUBSYSTEMS

All the required power for the ordnance disposal rover is supplied form a Lead-Acid Battery with the rating of 12V at 1.5 Amperes and with the capacity of 1500 mAH. We have used a step-down converter known as buck converter to divide voltage into 5V and 12V.

The DC-DC Buck Converter Step Down Module LM2596 Power Supply is a step-down(buck) switching regulator, capable of driving a 3-A load with excellent line and load regulation. These devices are available in fixed output voltages of 3.3 V, 5 V, 12 V, and an adjustable output version. The LM2596 series operates at a switching frequency of 150kHz, thus allowing smaller sized filter components than what would be required with lower frequency switching regulators.

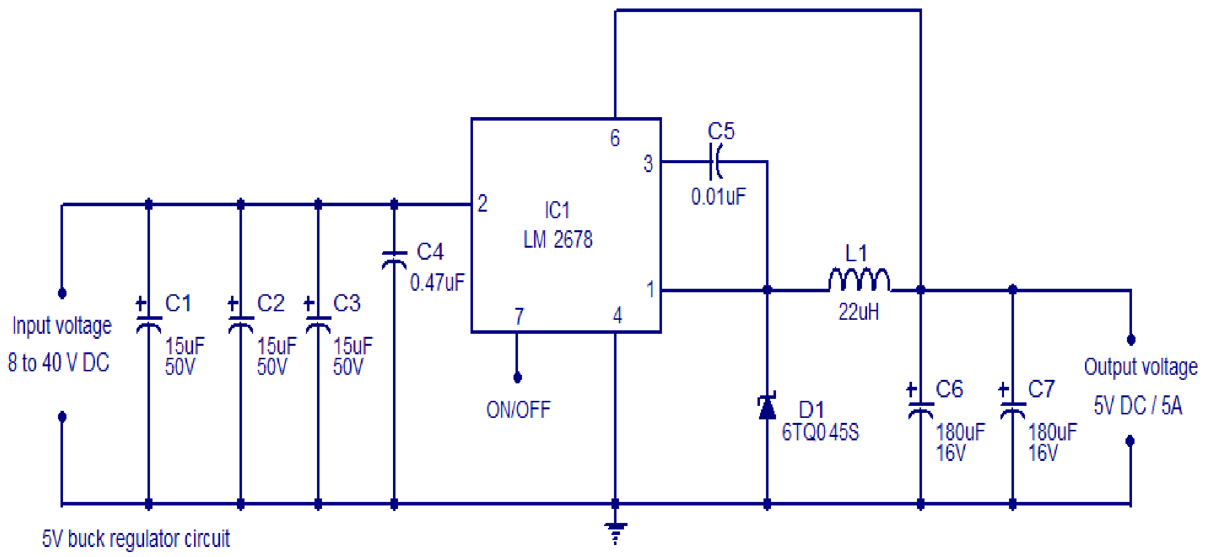


FIGURE 4.10 5V BUCK CONVERTER CIRCUIT

The 12V power supply is directly used by L298 motor shield for driving the two 12V DC motors.

The 5V is used by 3 modules:

1. Raspberry Pi via a 5V to USB Converter which converts input amperes form 1.5A to 2A which is ideal for a USB connection.
2. Relay Module for powering the lights and siren.
3. L298 Motor Shield for Motor driver circuit function.

All the Ground connections from the motor shield, buck converter, relay, and Lead-Acid Battery are connected to the Ground of Raspberry Pi’s GPIO Ground connections.

4.2.5 RELAY CIRCUIT:

A relay is an electromagnetic switch operated by a relatively small electric current that can turn on or off a much larger electric current. t works on the principle of an electromagnetic attraction. When the circuit of the relay senses the fault current, it energizes the electromagnetic field which produces the temporary magnetic field. This magnetic field moves the relay armature for opening or closing the connections. The small power relay has only one contacts, and the high-power relay has two contacts for opening the switch. The inner section of the relay is shown in the figure below. It has an iron core which is wound by a control coil. The power supply is given to the coil through the contacts of the load and the control switch. The current flows through the coil produces the magnetic field around it. Due to this magnetic field, the upper arm of the magnet attracts the lower arm. Hence close the circuit, which makes the current flow through the load. If the contact is already closed, then it moves oppositely and hence open the contacts. Here we use the relay to turn on and off the lights and sirens in the rover.

The relay has two different types of electrical contacts inside – normally open (NO) and normally closed (NC). The one you use will depend on whether you want the 5V signal to turn the switch on or turn the switch off. The 120-240V supply current enters the relay at the common (C) terminal in both configurations. To use the normally open contacts, use the NO terminal. To use the normally closed contacts, use the NC terminal.

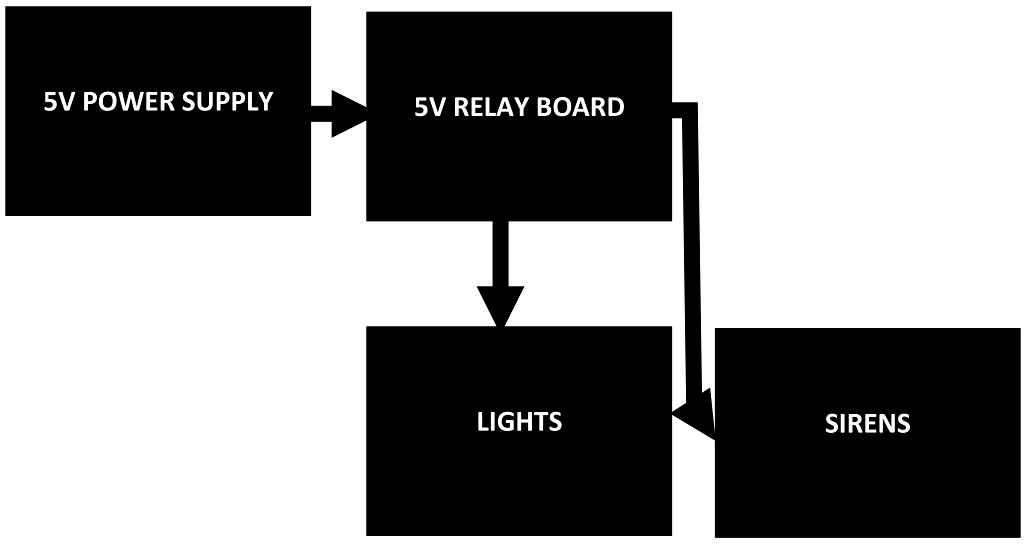


FIGURE 4.11 RELAY SUBSYSTEMS

4.2.6 Raspberry Pi Camera Module:

The Raspberry Pi Camera Module is an official product from the Raspberry Pi Foundation.

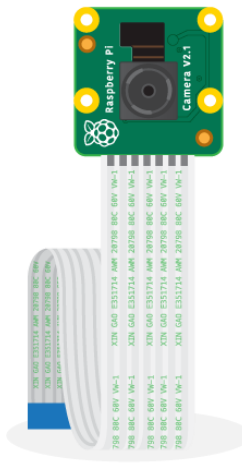


FIGURE 4.12 RASPBERRY PI CAMERA MODULE V1.3

The Raspberry Pi Camera Board plugs directly into the CSI connector on the Raspberry Pi. It's able to deliver a crystal clear 5MP resolution image, or 1080p HD video recording at 30fps. The module attaches to Raspberry Pi, by way of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI), which was designed especially for interfacing to cameras .

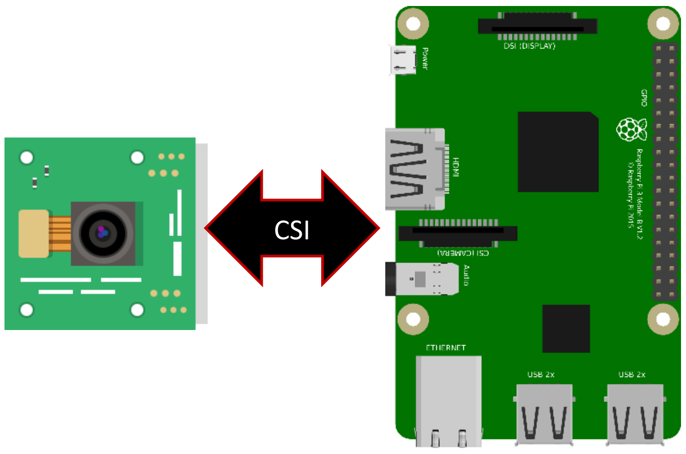


FIGURE 4.13 RASPBERRY PI CAMERA SERIAL INTERFACE

CHAPTER 5
REQUIRED INSTALLATIONS AND LIBRARIES

5.1 INSTALLATION OF RASPBIAN OS:

Raspbian is our official Debian optimized operating system for all models of the Raspberry Pi. You need to download 2 software and 1 OS i.e. Raspbian for this complete process.

1. The first software is Win32 Disk Imager.
   https://sourceforge.net/projects/win32diskimager/
2. Second software is SD Card Formatter.
   https://www.sdcard.org/downloads/formatter_4/
3. Raspbian OS: Download latest version for the Pi.
   https://www.raspberrypi.org/downloads/raspbian/

STEPS:

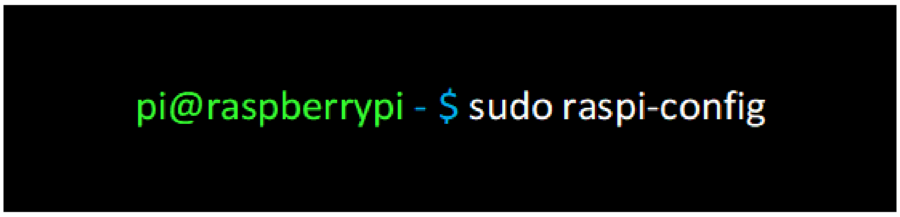
1. Get a minimum 8GB class 10 SD card with a card reader. Insert that card into the card reader and plug that to the USB port.
2. Open SD Card Formatter and select the drive you noticed in the previous step. Click on format and don't alter any other options. When formatting is completed, click on OK.
3. Open win32diskimager. Browse the .img file of Raspbian OS that was extracted from the downloaded file. Click on open and then click on Write. Wait for the write to be completed and it may take some minutes. So be patient.
4. Eject SD Card and insert it in Raspberry Pi Board.

5.2 CONFIGURING OF RASPBIAN OS:

In order to configure Raspbian OS on Raspberry Pi Board, we need a peripheral external keyboard and mouse to follow the below procedures. After once setup, we can use a VNC Server to establish connection between Computer and Raspberry Pi.

5.2.1 RASPBERRY PI CONFIGURATION:

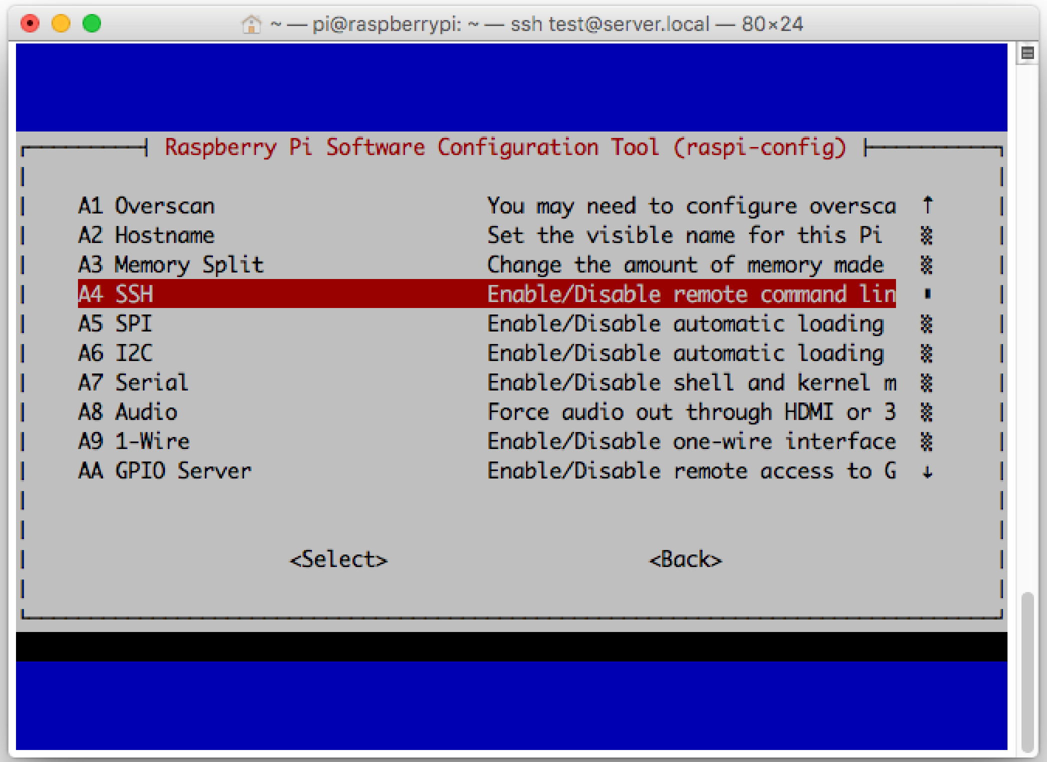
Connect Raspberry Pi to your HDMI monitor or TV. Put a keyboard and mouse into Raspberry Pi USB ports.



```
pi@raspberrypi - $ sudo raspi-config
```

Use the above command to open up Raspberry Pi Software Configuration Tool. Enable:

1. P1 Camera
2. P2 SSH
3. P3 VNC

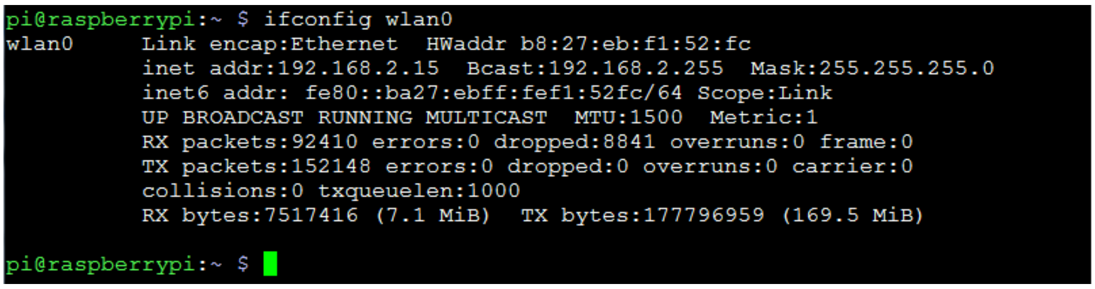


```
~ — pi@raspberrypi: ~ — ssh test@server.local — 80×24

┌─────────┤ Raspberry Pi Software Configuration Tool (raspi-config) ├──────────┐
│                                                                              │
│     A1 Overscan                      You may need to configure oversca ↑     │
│     A2 Hostname                      Set the visible name for this Pi  ▒     │
│     A3 Memory Split                  Change the amount of memory made  ▒     │
│     A4 SSH                           Enable/Disable remote command lin ▮     │
│     A5 SPI                           Enable/Disable automatic loading  ▒     │
│     A6 I2C                           Enable/Disable automatic loading  ▒     │
│     A7 Serial                        Enable/Disable shell and kernel m ▒     │
│     A8 Audio                         Force audio out through HDMI or 3 ▒     │
│     A9 1-Wire                        Enable/Disable one-wire interface ▒     │
│     AA GPIO Server                   Enable/Disable remote access to G ↓     │
│                                                                              │
│                                                                              │
│                     <Select>                     <Back>                      │
│                                                                              │
└──────────────────────────────────────────────────────────────────────────────┘
```

FIGURE 5.1 RASPBERRY PI SOFTWARE CONFIGURATION TOOL

5.2.2 WLAN CONFIGURATION: Open terminal and type the following to able to see your IP ADDRESS.

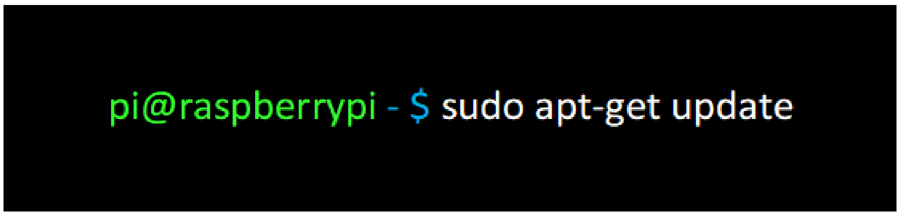


```
pi@raspberrypi:~ $ ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr b8:27:eb:f1:52:fc
          inet addr:192.168.2.15  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::ba27:ebff:fef1:52fc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:92410 errors:0 dropped:8841 overruns:0 frame:0
          TX packets:152148 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7517416 (7.1 MiB)  TX bytes:177796959 (169.5 MiB)

pi@raspberrypi:~ $
```

FIGURE 5.2 WLAN CONFIGURATION

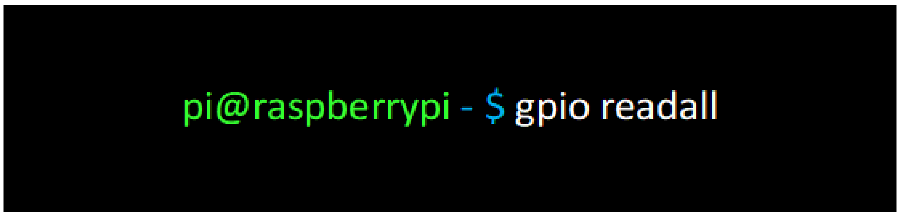
5.2.3 UPDATE LIBRARY: Update Raspbian Repository by typing following terminal command.



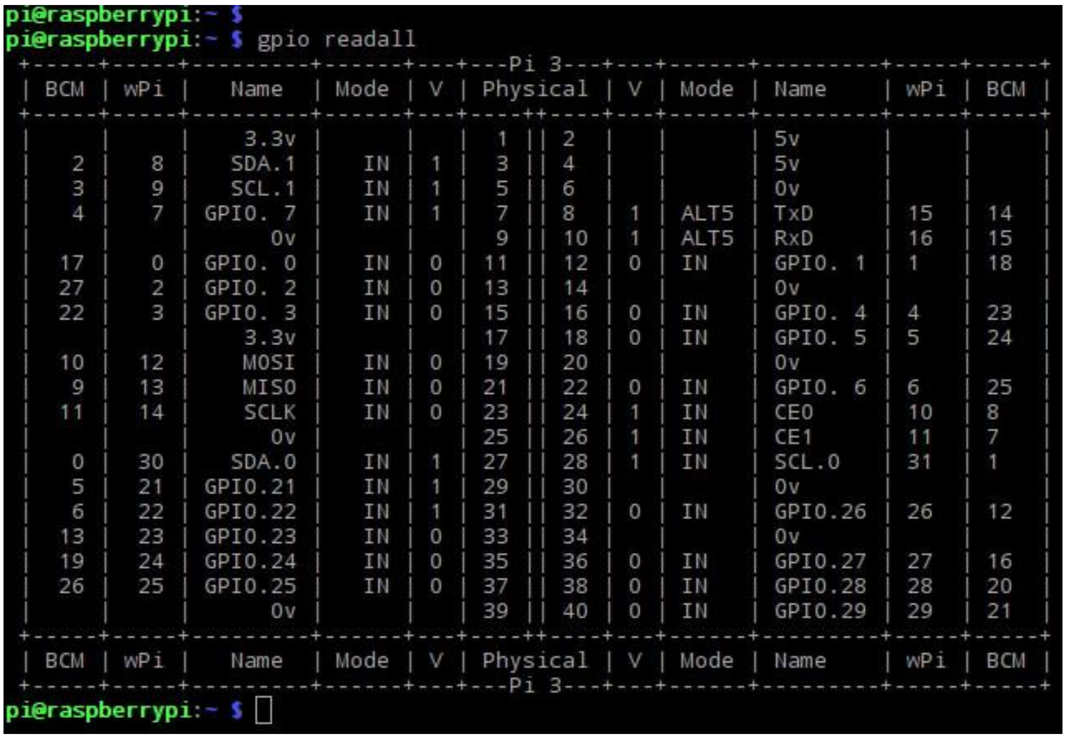
```
pi@raspberrypi - $ sudo apt-get update
```

This will update your GPIO Libraries, WiringPi, Python Scripts, Github, and rest of the installed software present on the OS.

5.2.4 CONFIGURE GPIO's: To read all the normally accessible pins and prints a table of their numbers you can use the command below, that create a cross-reference chart, with their modes and current values. This command also will detect the version/model of your RPi and printout the pin diagram appropriate to your Pi.



```
pi@raspberrypi - $ gpio readall
```



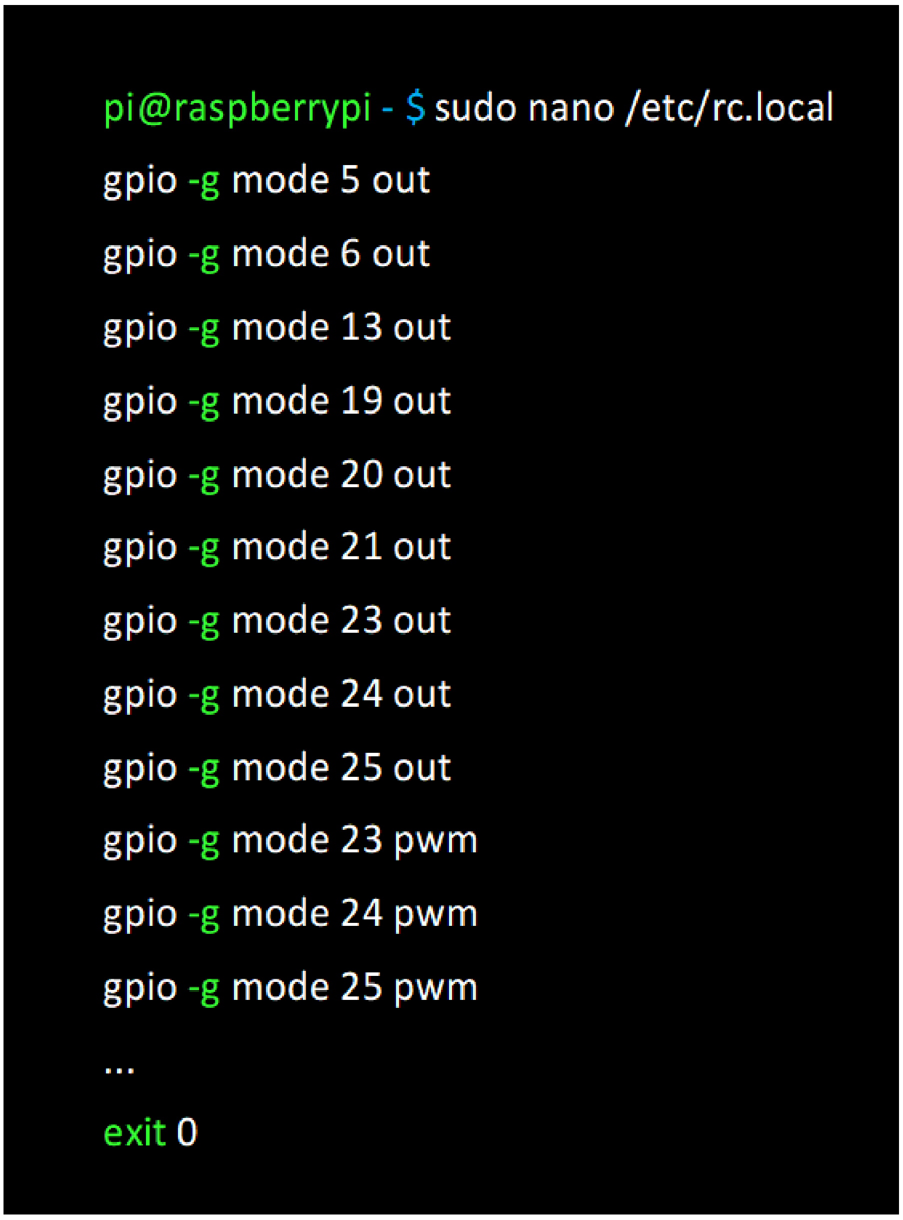
```
pi@raspberrypi:~ $
pi@raspberrypi:~ $ gpio readall
 +-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 |     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
 |   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
 |   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
 |   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT5 | TxD     | 15  | 14  |
 |     |     |      0v |      |   |  9 || 10 | 1 | ALT5 | RxD     | 16  | 15  |
 |  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
 |  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
 |  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
 |     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
 |  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
 |   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
 |  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
 |     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     | 11  | 7   |
 |   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
 |   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
 |   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
 |  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
 |  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
 |  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
 |     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
 +-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
 | BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
 +-----+-----+---------+------+---+---Pi 3---+---+------+---------+-----+-----+
pi@raspberrypi:~ $
```

FIGURE 5.3 GPIO READALL CHART

BCM	MODE	PURPOSE
5	OUT	DC MOTOR CONTROL USING L298 MOTOR SHIELD
6	OUT	
13	OUT	
19	OUT	
20	OUT	USED FOR RELAY CONNECTION
21	OUT	
23	OUT	PWM MODE FOR CONTROLLING ROBOTIC ARM SERVOS.
24	OUT	
25	OUT	

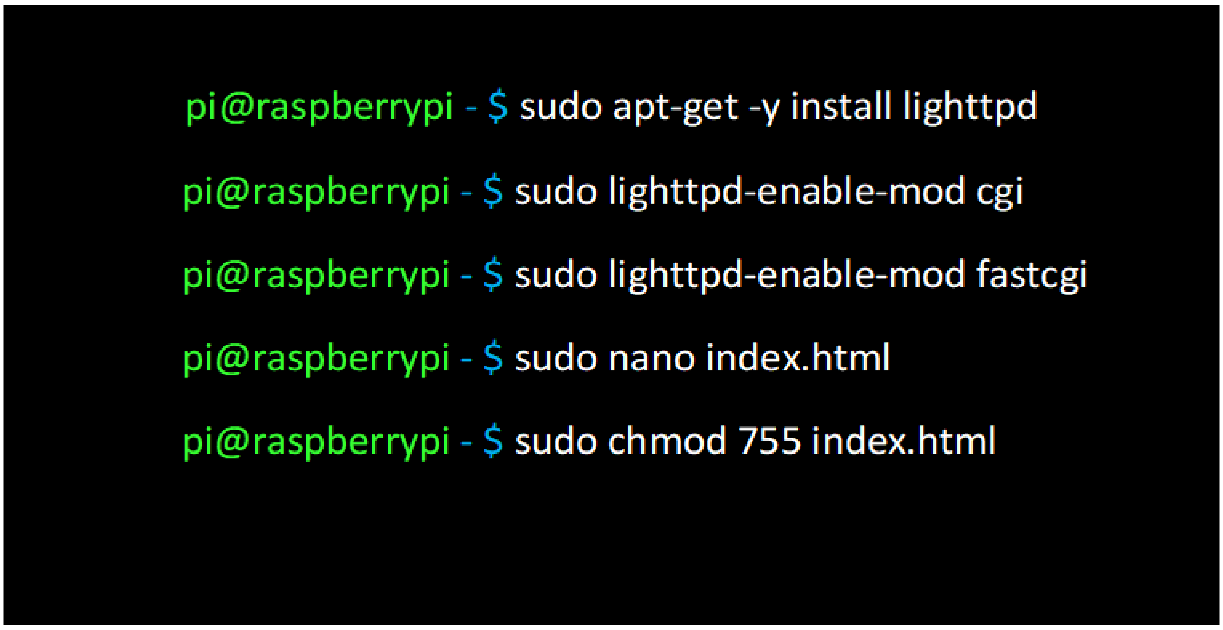
TABLE 5.1 GPIO REQUIRED MODES

If you reboot the RPi, the GPIOs will return to their default condition that is INPUT. So, we must change the script that is executed at any Raspberry's start. Here we need to define which GPIOs will be used and change the respective pin mode to OUT by using the following command:



```
pi@raspberrypi - $ sudo nano /etc/rc.local
gpio -g mode 5 out
gpio -g mode 6 out
gpio -g mode 13 out
gpio -g mode 19 out
gpio -g mode 20 out
gpio -g mode 21 out
gpio -g mode 23 out
gpio -g mode 24 out
gpio -g mode 25 out
gpio -g mode 23 pwm
gpio -g mode 24 pwm
gpio -g mode 25 pwm
...
exit 0
```

5.2.5 WEBSERVER: We will install LIGHTTPD, it is a secure, fast, compliant, and very flexible web-server that has been optimized for high-performance environments. It has a very low memory footprint and takes care of CPU-load. LIGHTTPD is free and open-source software and is distributed under the BSD license. It runs natively on Unix-like operating systems, as well as Microsoft Windows. LIGHTTPD supports the Fast CGI, SCGI and CGI interfaces to external programs, allowing web applications written in any programming language to be used with the server.



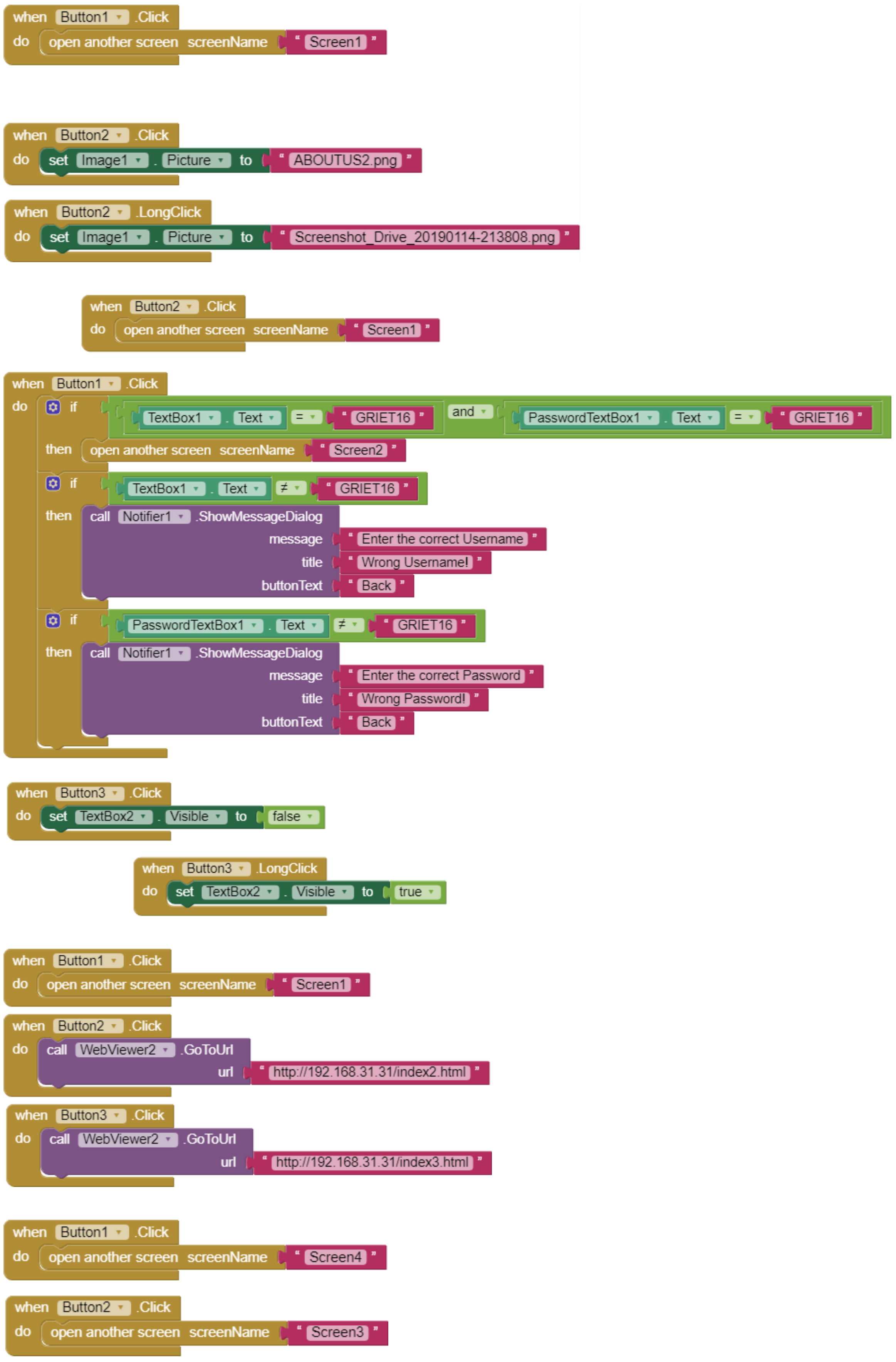
```
pi@raspberrypi - $ sudo apt-get -y install lighttpd
pi@raspberrypi - $ sudo lighttpd-enable-mod cgi
pi@raspberrypi - $ sudo lighttpd-enable-mod fastcgi
pi@raspberrypi - $ sudo nano index.html
pi@raspberrypi - $ sudo chmod 755 index.html
```

We edit index.html to the requirements for the project. This will be displayed on the RPi’s IP ADDRESS (We had set it to STATIC IP ADDRESS to 192.168.31.31)

5.2.6 MOBILE APPLICATION: We use MIT APP INVENTOR 2 for compiling an Android Application. App Inventor lets you develop applications for Android phones using a web browser and either a connected phone or emulator. The App Inventor servers store your work and help you keep track of your projects. Your app appears on the phone step-by-step as you add pieces to it, so you can test your work as you build.
You build apps with two major steps:

1. **The App Inventor Designer**, where you select the components for your app.
2. **The App Inventor Blocks Editor**, where you assemble program blocks that specify how the components should behave. You assemble programs visually, fitting pieces together like pieces of a puzzle.

When you're done, you can package your app and produce a stand-alone android application known as EODROVER.APK to install it. Give the application all the necessary permissions required to function correctly.



```
when Button1 .Click
do  open another screen screenName  " Screen1 "

when Button2 .Click
do  set Image1 . Picture to  " ABOUTUS2.png "

when Button2 .LongClick
do  set Image1 . Picture to  " Screenshot_Drive_20190114-213808.png "

when Button2 .Click
do  open another screen screenName  " Screen1 "

when Button1 .Click
do  if    TextBox1 . Text  =  " GRIET16 "  and  PasswordTextBox1 . Text  =  " GRIET16 "
    then  open another screen screenName  " Screen2 "
    if    TextBox1 . Text  ≠  " GRIET16 "
    then  call Notifier1 .ShowMessageDialog
                  message    " Enter the correct Username "
                  title      " Wrong Username! "
                  buttonText " Back "
    if    PasswordTextBox1 . Text  ≠  " GRIET16 "
    then  call Notifier1 .ShowMessageDialog
                  message    " Enter the correct Password "
                  title      " Wrong Password! "
                  buttonText " Back "

when Button3 .Click
do  set TextBox2 . Visible to  false

when Button3 .LongClick
do  set TextBox2 . Visible to  true

when Button1 .Click
do  open another screen screenName  " Screen1 "

when Button2 .Click
do  call WebViewer2 .GoToUrl
                url  " http://192.168.31.31/index2.html "

when Button3 .Click
do  call WebViewer2 .GoToUrl
                url  " http://192.168.31.31/index3.html "

when Button1 .Click
do  open another screen screenName  " Screen4 "

when Button2 .Click
do  open another screen screenName  " Screen3 "
```

FIGURE 5.4 APP INVENTOR SCREEN BLOCKS

CHAPTER 6
FIRMWARE DEVELOPMENT

6.1 PYTHON:

We have used the Python language has it has a diversified application in the software development and it is very easy to use on Raspberry PI. Python IDLE is already pre-installed with Debian based Raspbian Operating System. We have used a program called IPython is an interactive Python shell with syntax highlighting, autocompletion, pretty printing, built-in documentation, and more. IPython is not installed by default. Install via terminal or PUTTY with:

```
pi@raspberrypi - $ sudo pip3 install ipython
```

We have created two python scripts for functioning of our project model. The first program called “CAMERA LIVE STREAM.py” is used to live stream the video content from the attached raspberry pi camera via web interface. The second program called “MOTOR CONTROL.py” is used to control the movements of the motors present on the chassis, i.e. , movement of track wheels from the two DC motors and movement of robotic arm’s three servo motors. To run the programs, we have to run them via terminal,

```
pi@raspberrypi - $ sudo python3 CAMERA LIVE STREAM.py
pi@raspberrypi - $ sudo python3 MOTOR CONTROL.py
```

6.1.1: CAMERA LIVE STREAM:

```
import sys
import io
import os
import shutil
from subprocess import Popen, PIPE
from string import Template
from struct import Struct
from threading import Thread
from time import sleep, time
from http.server import HTTPServer, BaseHTTPRequestHandler
from wsgiref.simple_server import make_server
import picamera
from ws4py.websocket import WebSocket
from ws4py.server.wsgirefserver import (
    WSGIServer,
    WebSocketWSGIHandler,
    WebSocketWSGIRequestHandler,
)
from ws4py.server.wsgiutils import WebSocketWSGIApplication

###############################################CONFIGURATION

WIDTH = 640
HEIGHT = 480
FRAMERATE = 24
HTTP_PORT = 8082
WS_PORT = 8084
COLOR = u'#444'
BGCOLOR = u'#333'
JSMPEG_MAGIC = b'jsmp'
JSMPEG_HEADER = Struct('>4sHH')
VFLIP = False
```

```
HFLIP = False

###########################################

class StreamingHttpHandler(BaseHTTPRequestHandler):
    def do_HEAD(self):
        self.do_GET()
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
            return
        elif self.path == '/jsmpg.js':
            content_type = 'application/javascript'
            content = self.server.jsmpg_content
        elif self.path == '/index.html':
            content_type = 'text/html; charset=utf-8'
            tpl = Template(self.server.index_template)
            content = tpl.safe_substitute(dict(
                WS_PORT=WS_PORT, WIDTH=WIDTH, HEIGHT=HEIGHT,
COLOR=COLOR,
                BGCOLOR=BGCOLOR))
        else:
            self.send_error(404, 'File not found')
            return
        content = content.encode('utf-8')
        self.send_response(200)
        self.send_header('Content-Type', content_type)
        self.send_header('Content-Length', len(content))
        self.send_header('Last-Modified', self.date_time_string(time()))
        self.end_headers()
        if self.command == 'GET':
            self.wfile.write(content)
```

```
class StreamingHttpServer(HTTPServer):
    def __init__(self):
        super(StreamingHttpServer, self).__init__(
                ('', HTTP_PORT), StreamingHttpHandler)
        with io.open('index.html', 'r') as f:
            self.index_template = f.read()
        with io.open('jsmpg.js', 'r') as f:
            self.jsmpg_content = f.read()


class StreamingWebSocket(WebSocket):
    def opened(self):
        self.send(JSMPEG_HEADER.pack(JSMPEG_MAGIC, WIDTH, HEIGHT),
binary=True)


class BroadcastOutput(object):
    def __init__(self, camera):
        print('Spawning background conversion process')
        self.converter = Popen([
            'ffmpeg',
            '-f', 'rawvideo',
            '-pix_fmt', 'yuv420p',
            '-s', '%dx%d' % camera.resolution,
            '-r', str(float(camera.framerate)),
            '-i', '-',
            '-f', 'mpeg1video',
            '-b', '800k',
            '-r', str(float(camera.framerate)),
            '-'],
            stdin=PIPE, stdout=PIPE, stderr=io.open(os.devnull, 'wb'),
            shell=False, close_fds=True)

    def write(self, b):
        self.converter.stdin.write(b)

    def flush(self):
        print('Waiting for background conversion process to exit')
        self.converter.stdin.close()
        self.converter.wait()


class BroadcastThread(Thread):
    def __init__(self, converter, websocket_server):
        super(BroadcastThread, self).__init__()
        self.converter = converter
        self.websocket_server = websocket_server

    def run(self):
        try:
            while True:
                buf = self.converter.stdout.read1(32768)
                if buf:
                    self.websocket_server.manager.broadcast(buf, binary=True)
                elif self.converter.poll() is not None:
                    break
        finally:
            self.converter.stdout.close()


def main():
    print('Initializing camera')
    with picamera.PiCamera() as camera:
        camera.resolution = (WIDTH, HEIGHT)
        camera.framerate = FRAMERATE
        camera.vflip = VFLIP # flips image rightside up, as needed
        camera.hflip = HFLIP # flips image left-right, as needed
        sleep(1) # camera warm-up time
```

```
print('Initializing websockets server on port %d' % WS_PORT)
WebSocketWSGIHandler.http_version = '1.1'
websocket_server = make_server(
    '', WS_PORT,
    server_class=WSGIServer,
    handler_class=WebSocketWSGIRequestHandler,
    app=WebSocketWSGIApplication(handler_cls=StreamingWebSocket))
websocket_server.initialize_websockets_manager()
websocket_thread = Thread(target=websocket_server.serve_forever)

print('Initializing HTTP server on port %d' % HTTP_PORT)
http_server = StreamingHttpServer()
http_thread = Thread(target=http_server.serve_forever)
print('Initializing broadcast thread')
output = BroadcastOutput(camera)
broadcast_thread = BroadcastThread(output.converter, websocket_server)
print('Starting recording')
camera.start_recording(output, 'yuv')
try:
    print('Starting websockets thread')
    websocket_thread.start()
    print('Starting HTTP server thread')
    http_thread.start()
    print('Starting broadcast thread')
    broadcast_thread.start()
    while True:
        camera.wait_recording(1)
except KeyboardInterrupt:
    pass
finally:
    print('Stopping recording')
    camera.stop_recording()
    print('Waiting for broadcast thread to finish')
    broadcast_thread.join()
```

```
        print('Shutting down HTTP server')
        http_server.shutdown()
        print('Shutting down websockets server')
        websocket_server.shutdown()
        print('Waiting for HTTP server thread to finish')
        http_thread.join()
        print('Waiting for websockets thread to finish')
        websocket_thread.join()


if __name__ == '__main__':
    main()
```

6.1.1 MOTOR CONTROL PROGRAM:

```
import RPi.GPIO as GPIO
from time import sleep
import cgi, cgit

in1 = 05
in2 = 06
in3 = 13
in4 = 19
arm1 = 16
arm2 = 18
arm3 = 22

GPIO.setmode(GPIO.BCM)
GPIO.setup(in1,GPIO.OUT)
GPIO.setup(in2,GPIO.OUT)
GPIO.setup(in3,GPIO.OUT)
GPIO.setup(in4,GPIO.OUT)
GPIO.setup(arm1,GPIO.OUT)
GPIO.setup(arm2,GPIO.OUT)
GPIO.setup(arm3,GPIO.OUT)
GPIO.output(in1,GPIO.LOW)
GPIO.output(in2,GPIO.LOW)
GPIO.output(in3,GPIO.LOW)
GPIO.output(in4,GPIO.LOW)
GPIO.output(arm1,GPIO.LOW)
GPIO.output(arm2,GPIO.LOW)
GPIO.output(arm3,GPIO.LOW)
p=GPIO.PWM(en,1000)
q=GPIO.PWM(en,1000)
p.start(25)
q.start(50)


while(1):
            x=raw_input()

    if x=='s':
        print("stop")
        GPIO.output(in1,GPIO.LOW)
        GPIO.output(in2,GPIO.LOW)
        GPIO.output(in3,GPIO.LOW)
        GPIO.output(in4,GPIO.LOW)
        x='z'
        exec(open("/var/www/cgi-bin/stop.cgi").read())

    elif x=='fowr':
        print("forward")
        GPIO.output(in1,GPIO.HIGH)
        GPIO.output(in2,GPIO.LOW)
        GPIO.output(in3,GPIO.HIGH)
        GPIO.output(in4,GPIO.LOW)
        x='z'
        exec(open("/var/www/cgi-bin/forward.cgi").read())
```

```
elif x=='back':
    print("backward")
    GPIO.output(in1,GPIO.LOW)
    GPIO.output(in2,GPIO.HIGH)
    GPIO.output(in3,GPIO.LOW)
    GPIO.output(in4,GPIO.HIGH)
    x='z'
    exec(open("/var/www/cgi-bin/reverse.cgi").read())

elif x=='left':
    print("left")
    GPIO.output(in1,GPIO.HIGH)
    GPIO.output(in2,GPIO.LOW)
    GPIO.output(in3,GPIO.LOW)
    GPIO.output(in4,GPIO.HIGH)
    x='z'
    exec(open("/var/www/cgi-bin/left.cgi").read())

elif x=='right':
    print("right")
    GPIO.output(in1,GPIO.LOW)
    GPIO.output(in2,GPIO.HIGH)
    GPIO.output(in3,GPIO.HIGH)
    GPIO.output(in4,GPIO.LOW)
   x='z'
 exec(open("/var/www/cgi-bin/right.cgi").read())

elif x=='lowspeed':
    print("low")
    p.ChangeDutyCycle(25)
    x='z'

elif x=='mediumspeed':
    print("medium")
    p.ChangeDutyCycle(50)
    x='z'

elif x=='highspeed':
    print("high")
    p.ChangeDutyCycle(75)
    x='z'

elif x=='centertilt':
    GPIO.output(arm1,GPIO.HIGH)
    print("center")
    q.ChangeDutyCycle(5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/centertilt.cgi").read())

elif x=='righttilt':
    GPIO.output(arm1,GPIO.HIGH)
    print("right tilt")
    q.ChangeDutyCycle(2.5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/righttilt.cgi").read())

elif x=='lefttilt':
    GPIO.output(arm1,GPIO.HIGH)
    print("lef tilt")
    q.ChangeDutyCycle(7.5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/lefttilt.cgi").read())

elif x=='centerpan':
    GPIO.output(arm2,GPIO.HIGH)
    print("center pan")
    q.ChangeDutyCycle(5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/centerpan.cgi").read())

elif x=='rightpan':
    GPIO.output(arm2,GPIO.HIGH)
    print("right pan")
    q.ChangeDutyCycle(2.5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/rightpan.cgi").read())

elif x=='leftpan':
    GPIO.output(arm2,GPIO.HIGH)
    print("left pan")
    q.ChangeDutyCycle(7.5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/leftpan.cgi").read())

elif x=='armopen':
    GPIO.output(arm3,GPIO.HIGH)
    print("arm open")
    q.ChangeDutyCycle(0.5)
    time.sleep(1)
    x='z'
    exec(open("/var/www/cgi-bin/openarm.cgi").read())

elif x=='armclose':
    GPIO.output(arm3,GPIO.HIGH)
    print("arm close")
```

```
        q.ChangeDutyCycle(2.5)
        time.sleep(1)
        x='z'
        exec(open("/var/www/cgi-bin/closearm.cgi").read())

 elif x=='clean':
        print("GPIOs have been cleaned")
        GPIO.cleanup()
        break
```

6.2 HYPERTEXT MARKUP LANGUAGE:

We edit index.html to the requirements for the project. This will be displayed on the RPi's IP ADDRESS. We had set the IP ADDRESS to default STATIC. Now the default IP ADDRESS is 192.168.31.31.

6.2.1 INDEX.HTML:

```
<html>
<style>
body {background-color: black}
h1 {color:red}
p {color:red}
a {color:red}
button {
    color:red;
    background:rgb(0, 0, 0);
    border: 1px solid red;
    border-radius: 8px;
    position: center;
}
</style></head>
<body data-gr-c-s-loaded="true">
<div style="text-align:center">
```

```
<a href="index1.html"> Live Camera </a>
<br>
<iframe src="http://192.168.31.31:9000/javascript_simple.html" frameborder="0"
align="center" width="340" height="260" scrolling="no" ></iframe>
<p>Directions</p>
<button style="height: 75px; width: 75px" onclick="lighton()"><img style="height: 50px"
src="/images/lighton.png"></button>
<button style="height: 75px; width: 75px" onclick="forward()"><img style="height: 50px"
src="./images/forward.png"></button>
<button style="height: 75px; width: 75px" onclick="lightoff()"><img style="height: 50px"
src="/images/lightoff.png"></button>
<br>
<button style="height: 75px; width: 75px" onclick="left()"><img style="height: 50px"
src="/images/left.png"></button>
<button style="height: 75px; width: 75px" onclick="stop()"><img style="height: 50px"
src="/images/stop.png"></button>
<button style="height: 75px; width: 75px" onclick="right()"><img style="height: 50px"
src="/images/right.png"></button>
<br>
<button style="height: 75px; width: 75px" onclick="redlighton()"><img style="height: 50px"
src="/images/redlighton.png"></button>
<button style="height: 75px; width: 75px" onclick="reverse()"><img style="height: 50px"
src="/images/reverse.png"></button>
<button style="height: 75px; width: 75px" onclick="redlightoff()"><img style="height:
50px" src="/images/redlightoff.png"></button>
<br>
<p>Tilt Angle</p>
 <img hspace="18" style="padding-left: 5px">
 <button style="height: 50px; width: 50px; font-size: 25px" onclick="downtilt()">--</button>
 <button style="height: 50px; width: 50px; font-size: 25px" onclick="downcentertilt()">-
</button>
       <button style="height: 50px; width: 50px; font-size: 25px"
onclick="centertilt()">0</button>
```

```
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="upcentertilt()">+</button>
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="uptilt()">++</button>
        <img hspace="18" style="padding-left: 5px">
        <br>
 <p>Pan Position</p>
    <button style="height: 50px; width: 50px; font-size: 25px"
onclick="inpan()">++</button>
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="incenterpan()">I+</button>
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="centerpan()">C</button>
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="outcenterpan()">O+</button>
        <button style="height: 50px; width: 50px; font-size: 25px"
onclick="outpan()">++</button>
        <p>Claw</p>
        <button style="height: 50px; width: 135px; font-size: 25px"
onclick="clawopen()">Open</button>
        <button style="height: 50px; width: 135px; font-size: 25px"
onclick="clawclose()">Close</button>
 <p>
</p>
</div>
<script>
var xmlhttp;
xmlhttp=new XMLHttpRequest();

function lighton()
    {
        xmlhttp.open("GET","cgi-bin/redlighton.cgi",true);
        xmlhttp.send();
    }
```

```
function lightoff()
   {
      xmlhttp.open("GET","cgi-bin/redlightoff.cgi",true);
      xmlhttp.send();
   }
function forward()
{
   xmlhttp.open("GET","cgi-bin/left.cgi",true);
   xmlhttp.send();
}
function stop()
{
   xmlhttp.open("GET","cgi-bin/stop.cgi",true);
   xmlhttp.send();
}
function left()
{
   xmlhttp.open("GET","cgi-bin/forward.cgi",true);
   xmlhttp.send();
}
function right()
{
   xmlhttp.open("GET","cgi-bin/reverse.cgi",true);
   xmlhttp.send();
}
function reverse()
{
   xmlhttp.open("GET","cgi-bin/right.cgi",true);
   xmlhttp.send();
}
function downtilt()
{
   xmlhttp.open("GET","cgi-bin/downtilt.cgi",true);
   xmlhttp.send();
```

```
}
function downcentertilt()
{
   xmlhttp.open("GET","cgi-bin/downcentertilt.cgi",true);
   xmlhttp.send();
}
function centertilt()
{
   xmlhttp.open("GET","cgi-bin/centertilt.cgi",true);
   xmlhttp.send();
}
function upcentertilt()
{
   xmlhttp.open("GET","cgi-bin/upcentertilt.cgi",true);
   xmlhttp.send();
}
function uptilt()
{
   xmlhttp.open("GET","cgi-bin/uptilt.cgi",true);
   xmlhttp.send();
}
function inpan()
{
   xmlhttp.open("GET","cgi-bin/inpan.cgi",true);
   xmlhttp.send();
}
function incenterpan()
{
   xmlhttp.open("GET","cgi-bin/incenterpan.cgi",true);
   xmlhttp.send();
}
function centerpan()
{
   xmlhttp.open("GET","cgi-bin/centerpan.cgi",true);
   xmlhttp.send();
}function outcenterpan()
{
   xmlhttp.open("GET","cgi-bin/outcenterpan.cgi",true);
   xmlhttp.send();
}
function outpan()
{
   xmlhttp.open("GET","cgi-bin/outpan.cgi",true);
   xmlhttp.send();
}
function clawopen()
{
   xmlhttp.open("GET","cgi-bin/clawopen.cgi",true);
   xmlhttp.send();
}
function clawclose()
{
   xmlhttp.open("GET","cgi-bin/clawclose.cgi",true);
   xmlhttp.send();
}
function redlighton()
   {
      xmlhttp.open("GET","cgi-bin/lighton.cgi",true);
      xmlhttp.send();
   }
function redlightoff()
   {
      xmlhttp.open("GET","cgi-bin/lightoff.cgi",true);
      xmlhttp.send();
   }
</script>
</body>
</html>
```

6.3 OUTPUTS:

6.3.1 WEBPAGE: This is the output of index.html, this can be accessed from any web browser enable device such has computer, mobile phone, tablet, smart tv, and even smart watch

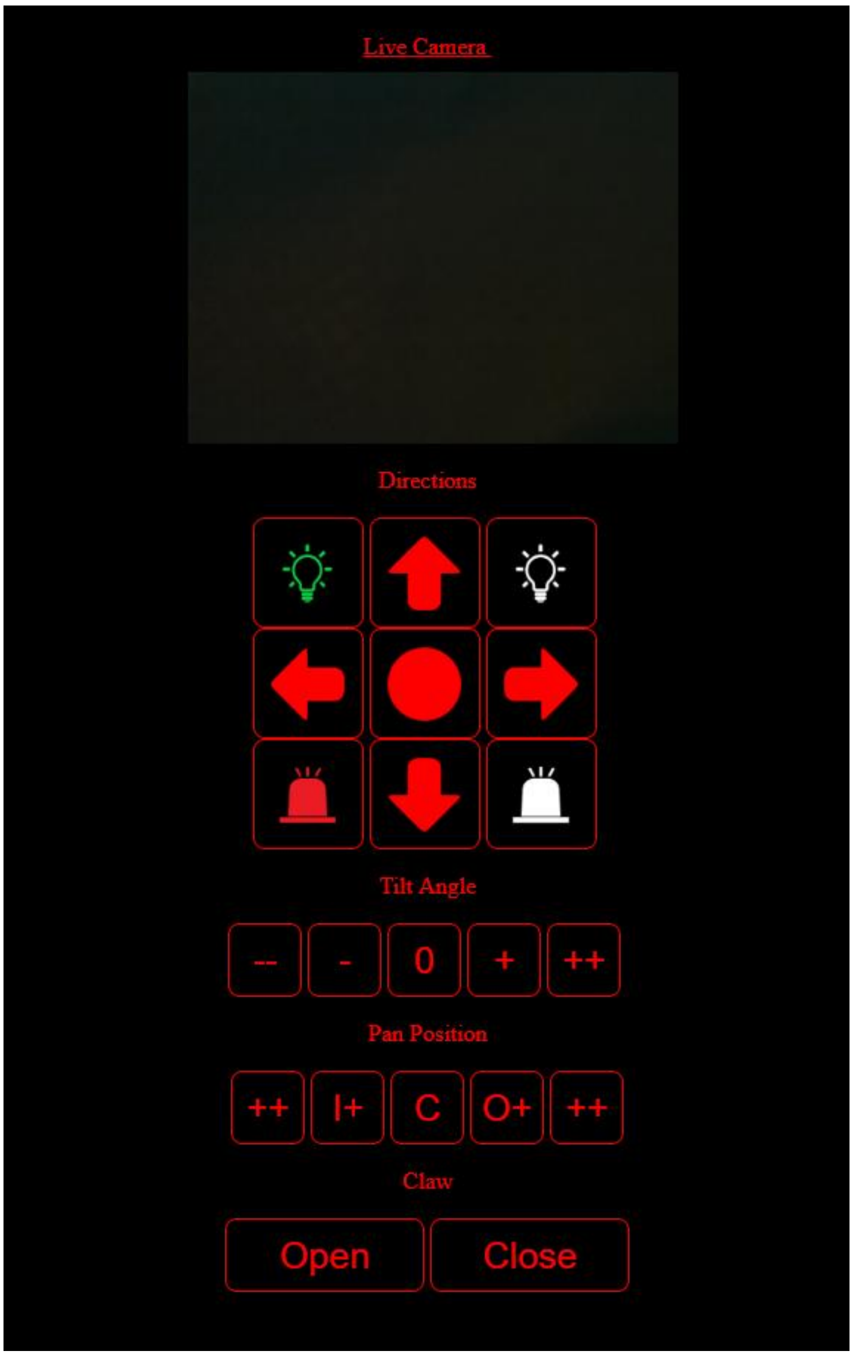
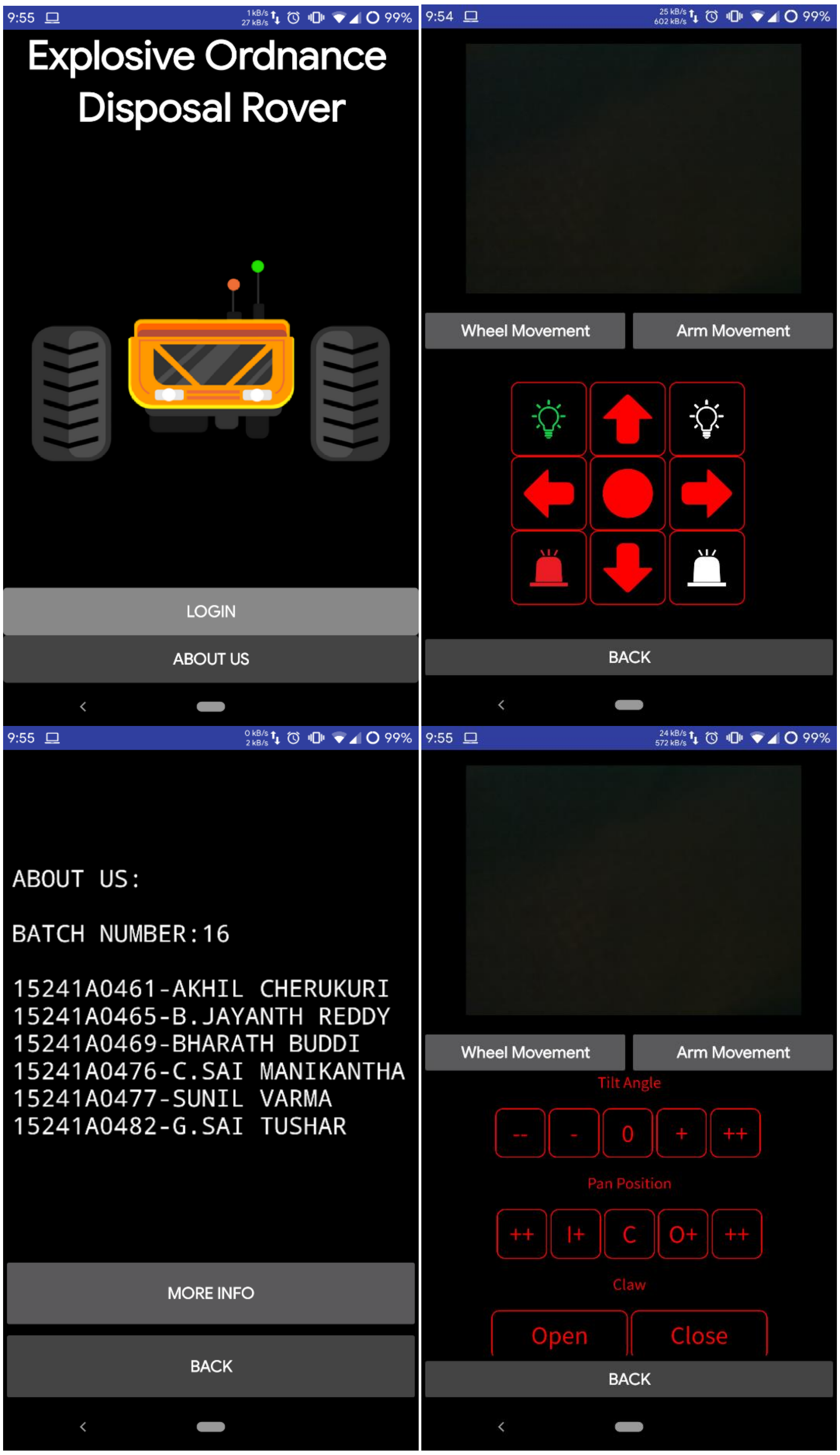


FIGURE 6.1 INDEX WEBPAGE

6.3.2 ANDROID APPLICATION:



ABOUT US:

BATCH NUMBER:16

15241A0461-AKHIL CHERUKURI
15241A0465-B.JAYANTH REDDY
15241A0469-BHARATH BUDDI
15241A0476-C.SAI MANIKANTHA
15241A0477-SUNIL VARMA
15241A0482-G.SAI TUSHAR

FIGURE 6.2 ANDROID SCREENSHOTS

CHAPTER 7
FUTURE ENHANCEMENTS

The system that we have built is a working prototype of a robot, which should be compact, fast and accurate. This prototype may not have the features and reliability of the original design. It is only being developed to ensure that the design is feasible, not impractical and can be implemented on a much larger scale in a more efficient way. At present, the robot is not a very maneuverable machine that is it may not provide the efficiency to cope with the complex objects or it may not have the capability to maneuver into small places, which is necessary requirement. But it can be used to design such a robot, which can be small in size, fast and accurate in its movements. The gripper as compared to the ones, made by professional companies is not very efficient. But it can still perform some level of object manipulation. Hence the future enhancements may include a much smaller, faster, more reliable machine. It may have the ability to handle a much wider range of objects and the ability to maneuver them to much safer places. Some of these enhancements are described below.

COMPACT DESIGN

A compact design results in a much faster motion and thus increases the accuracy and efficiency. Therefore, the robot can be enhanced to be of much smaller size for the purpose of a faster and accurate operation. Compact design is also required where the situation demands the robot to reach for small places. For example, in the aftermath of an earth quake, the robot has to search for people trapped under the rubble. It has to enter holes where humans cannot enter. Hence a compact robot will easily do the job.

QUICK MOVEMENT

Being a bomb disposal robot, it requires very fast movement. This is required as the bomb disposal squad have very little time in checking out the bomb and then defusing it. Therefore, a fast robot is necessary to be successfully used as an EXPLOSIVE ORDNANCE DISPOSAL ROVER.

IMPROVED RELIABILITY

At the moment the turning mechanism of the robot is based on the DC motor, which is not that accurate. Also, the shoulder of the robotic arm or the base of the robotic arm also depends on the motion of a DC motor, which is not very feasible for the accurate motion of the robotic arm, which is necessary. The robot can be improved to be more reliable and accurate by installing stepper motors instead of the DC motors, which are more accurate and also have the feature of the holding torque, which enables the movement of heavy object with ease.

REMOVABLE GRIPPER/MULTI-GRIPPER ROBOTIC ARM

The gripper attached to the robotic arm is fixed at the moment that is, it will only work with the specific shaped of objects. Placing a gripper that can be removed and replaced by another gripper can solve this problem or a multi gripper robotic arm can be developed with more than 2 types of grippers for different type of materials and for different shaped of the objects. This will enable the robotic arm to grip and move objects, which are complex and cannot be easily moved with a single or a basic gripper.

ARTIFICIAL INTELLIGENCE

At present, the robot cannot make decisions on its own that is there is no built-in artificial intelligence in it. Therefore, the robot's working is based purely on the decisions made by the end user of the robotic control application. Therefore, Artificial Intelligence may be provided to the robot for making the process of decision making much quicker and reliable. For example, a database can be merged in the application, which can be used to construct evaluation by identifying the object with the help of the camera. The camera gives input to the database that is compared with the contents of the database and if a match is found the corresponding entries of the database gives the desired actions to be performed by the robot on the object, which makes the task much easier for the end user.

Also, AI may provide a threat of performing wrong operations in case of identifying the object wrong which may result in a hazardous situation. Therefore, if Artificial Intelligence is being implemented then it should be made sure that the accuracy rate of the AI engine is almost 100% accurate otherwise it can be dangerous in such a situation.

NIGHT VISION CAMERA

The robot equipped with a wireless camera, which is not very useful in situations where the visibility or light level is very low. For night mode it will be almost impossible for identifying objects because the lights, which are provided on the robot, are fixed therefore it may not be possible to view those objects which are in the dark. For night mode or places where light is low a night vision camera can be mounted on the robot instead of a standard camera, which will increase the visibility in case of no light at all.

VIRTUAL REALITY

Previous generations of EOD robots depend on cameras, monitors, and joysticks. While functional, this arrangement is less than ideal, making the delicate task of disarming a bomb more challenging. The user just doesn't have as much control, and the bots are pretty clumsy in comparison with VR enabled robots. Moreover, the user, much like a drone pilot, simply sits in a chair in front of a monitor, controlling the robot at a physical and psychological distance.

VR EOD Robots designers adopted VR technology to create an immersive, immediate user interface. The explosives experts that use VR EOD Robots wear a VR headset and manipulate an VR Touch controller, putting themselves in the virtual place of the robot. In essence, the goal is for them to become the bot, to be there with the bomb. Some VR systems even allow tactile feedback through haptics, offering the user physical feedback when the robot touches something, the level of immersion and control is truly next-level.

LASER

EOD Rovers may soon use lasers to dispose of explosive devices. Lasers in the past few years have reduced the size from being a big module that produces a lot of heat and needed air conditioning to small hand-held lasers that also require less wattage to produce the needed laser beam. The plan is to position the lasers on top of a controlled robotic arm which can locate the explosive device. The laser then shoots a high intensity beam and to burn through the ordnance, and the idea is to burn through the outer case of it, and get the inside to not detonate, but to burn out.

CHAPTER 8
CONCLUSION

The **EXPLOSIVE ORDNANCE DISPOSAL ROVER** has been designed in such a way that it can cater to the needs of the bomb disposal squad, the military, the police and also for the personnel who handle radioactive materials. It has countless applications and can be used in different environments and scenarios. For instance, at one place it can be used by the bomb disposal squad, while at another instance it can be used for handling mines. While another application

In the past decade, robotic systems have been used with increased popularity for explosive ordnance (EOD) missions. Advances in robotic technology have made it possible for robots to per-form functions previously only possible by human workers wearing a blast suit as shown. Currently, EOD robots are able to traverse a variety of terrain, collect and destroy certain explosives and provide improved reconnaissance capabilities to law enforcement and military agencies. Although far from perfected, these robots are saving lives by finding and disposing of explosives without the need for direct human contact reliable robotic platform. The flexible and modular robot design utilizes commercial off the shelf components for ease of maintenance and repairs. The robot provides a safe distance threat assessment and increased capacity for explosive ordinance disposal, improving the effectiveness of bomb disposal teams. The robot's low-cost, intuitive operation and ease-of-maintenance promote its widespread appeal, thereby saving the lives of both law enforcement personnel and civilians. can be to provide up to date information in a hostage situation.

One of the major advantages of this robot is

- It can be altered to suit the needs of the user
- It is fast and robust.
- It can handle different loads.
- It can be controlled remotely.
- It has video feedback.
- It has its power supply.
- It has a 3-degree of freedom robotic arm.