

PROJECT REPORT ON
AN APPROACH FOR LANDMINE DETECTION AND SECURE
INFORMATION TRANSFER USING HIGH SENSITIVE AUTONOMOUS
ROBOT

Submitted in partial fulfillment of the requirements for the award of
the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
OF
SASTRA UNIVERSITY

Submitted by

N RADHIKA	117003154
M.S. JYOTHSSNA RINI	117003079
S.PRIYANKA	117003151



Under the Guidance of
Prof. KAMAKSHI .S
ASSISTANT PROFESSOR
SCHOOL OF COMPUTING
SASTRA UNIVERSITY, THANJAVUR

SCHOOL OF COMPUTING
SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613 401
April 2017

SCHOOL OF COMPUTING
SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401



BONAFIDE CERTIFICATE

Certified that this project work entitled “**AN APPROACH FOR LANDMINE DETECTION AND SECURE INFORMATION TRANSFER USING HIGH SENSITIVE AUTONOMOUS ROBOT**” submitted to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA University), Tirumalaisamudram-613401 by **N.RADHIKA (117003154), M.S. JYOTHSSNA RINI (117003079), S. PRIYANKA (117003151)**, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** is the original and independent work carried out under my guidance, during the period December 2016 - April 2017.

INTERNAL GUIDE
Prof. KAMAKSHI.S
SCHOOL OF COMPUTING

ASSOCIATE DEAN
Dr. A. UMAMAKESWARI
SCHOOL OF COMPUTING

Submitted for University Examination held on_____

EXAMINER – I

EXAMINER - II

**SCHOOL OF COMPUTING
SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401**



We submit this project work entitled “**AN APPROACH FOR LANDMINE DETECTION AND SECURE INFORMATION TRANSFER USING HIGH SENSITIVE AUTONOMOUS ROBOT**” to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA) University, Tirumalaisamudram–613 401, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** and declare that it is our original and independent work carried out under the guidance of **Prof. KAMAKSHI.S**, School of Computing, SASTRA.

Place : **Name:** N.RADHIKA Signature:

Date: **Reg.no:**117003154

Name: MS JYOTHSSNA RINI Signature:

Reg. no: 117003079

Name: PRIYANKA.S Signature:

Reg. No: 117003151

ACKNOWLEDGEMENT

First and Foremost, we take pride in thanking the almighty for giving us strength for the successful completion of this project.

We have immense pleasure in expressing our heartfelt thanks to our Vice-Chancellor **Prof. R. Sethuraman** for the benevolent advice and guidance during our tenure in the college.

We would like to express our gratitude to **Dr. G. Bhalachandran, Registrar**, SASTRA University, for his benevolent guidance through-out our college life.

We wish to express our thanks to **Dr. S.Vaidhyasubramanian, Dean, Planning and Development**, SASTRA University for his encouragement and providing us with all the needed amenities.

We would like to express our gratitude to **Dr .A. Umamakeshwari**, Associate Dean , Department of Computer Science and Engineering, School of Computing, SASTRA University, for her benevolent guidance through-out our college life.

We wish to express our thanks to **Prof. S. Kamakshi**, Assistant Professor, Department of Computer Science and Engineering, School of Computing, SASTRA University for her encouragement and providing us with all the needed amenities.

We deeply thank our family and friends for supporting us in all the tasks that we have carried for the successful completion of this project.

**N.RADHIKA
M.S. JYOTHSSNA RINI
S. PRIYANKA**

TABLE OF FIGURES

S.No.	Figure Title	Page No.
1	Command prompt of Raspberry Pi OS	10
2	Basic block diagram of the Proposed system	11
3	GPIO Pin Diagram for Landmine Detection Robot controlled web-based user	12
4	Components used	13
5	Various views of the Robot	13
6	Use case diagram for landmine detection robot	14
7	Activity diagram for landmine detection robot	16
8	Sequence diagram for landmine detection robot	19
9	Class diagram for landmine detection robot	21
10	Collaboration diagram for landmine detection robot	22
11	Output Screen of Web application controlling Robot	31
12	Method versus Time elapsed for various algorithms	32
13	Output Screen showing snippets of the main code execution	33
14	Shows mail sent from priyankasrs@gmail.com to jyorini30@gmail.com	34

TABLE OF CONTENTS

S.No	Chapter Title	Page No.
	Acknowledgment	iv
	Table of Figures	v
	Abstract	1
1.	Introduction	2
2.	Problem Statement	4
3.	Literature Survey	5
4.	Software and Hardware Requirement specification	6
	4.1 General hardware specification of Raspberry Pi 3	7
	4.2 Configuration of Raspberry Pi 3	8
	4.3 Required hardware components	9
	4.4 Software requirements specification	9
5.	Proposed architecture	11
6.	Interaction Scenario	14
	6.1 Use Case Diagram	16
	6.2 Activity Diagram	19
	6.3 Class Diagram	21
	6.4 Sequence Diagram	22
	6.5 Collaboration Diagram	22
7.	Methodology and Approach	23
8.	Output/Results	31
9.	Conclusion	35
10.	References	36
11.	Appendix	37

ABSTRACT

This project aims at the development of a web-based user interface to control and monitor the movement of Mobile Robot that is used for the purpose of Landmine Detection. The Robot is deployed in regions for military purposes and its movement is monitored through a remote web-based user interface. The major purpose is to detect the landmine and send an image of its location to the user. The robot is built with suitable materials to survive devastating situations like high temperatures and pressure. The built-in Wi-Fi in Raspberry Pi 3 helps to set up communication between the remote user interface and controller in the robot.

The process begins by providing inputs from web-based user interface as the predefined functions which have dedicated functionalities (Start & Stop).The Robot identifies these inputs in Raspberry Pi 3 and appropriate function is carried out. Actuators will receive signals from raspberry pi 3 and the tasks are performed. The web-based interface is used to monitor the robot which is interfaced with the landmine detecting sensors. The GPS module aids in guiding the Robot by providing proper localization. A metal detector is employed which helps in accurate detection of landmines buried under the soil. The IR sensors play a vital role in determining obstacles along the path of the Robot. On detection of landmine, the camera module captures the image of the present surroundings and an e-mail is generated through which the encrypted image of the location is sent to the registered e-mail id. For encryption, Integer Wavelet Transform (IWT) plus Play-Fair cipher technique is employed using Double - Level IWT – Encrypt LL1 and LL2 Bands (DLILL) method that sends the final encrypted image to the remote controller. Such an encryption scheme can keep the landmine location away from anti-social groups.

1. INTRODUCTION

Land mines are laid explosives found buried into the ground designed to injure or kill people. They can lie dormant or inactive for years until a person or an animal triggers their detonating mechanism. They are activated by pressure, by pulling a switch, by radio signal or any other remote firing method from a predetermined distance. The most alarming factor is that they indiscriminately injure civilians, soldiers or the mine clearance staff and hence are called “victim-activated”. While there are many activists working towards the ban in the manufacture and selling of these landmines, the real challenge lies in the accurate detection of around 100 million landmines buried in over 65 countries. These landmines are left behind after wars and have been an unsolved problem of the world and are a central issue in Europe, Africa, Asia and central South America. Not only do they affect the military domain but pose a serious threat to the agricultural sectors by making agricultural lands uncultivable. An effective solution for the problem must involve accurate detection of landmines with a minimal rate of false alarms without involving actual human interaction thus reducing the death rate caused in manual landmine detection. Secondly the information regarding the buried landmines must be transferred in a secure form to the concerned demining team so as to avoid the intervention of any other anti-social groups. One major objective should be the secure transfer of information mainly with regards to the image containing landmine location.

EXISTING SYSTEM:

In the systems that previously existed, dead reckoning or differential steer drive were used for navigation. It did not contribute much to render the Robot autonomous. It required explicit control to direct the movement of the Robot. Also, no previous system has employed an encryption technique for information transfer nor a mail generation system that dynamically updates the remote controller who controls the robot. The proposed system suggests a novel way of looking into this problem of landmine detection.

PROPOSED SYSTEM:

In the proposed system, we use Bug algorithm which is a path planning algorithm based on the concept of uninformed search for navigation of the robot. We also use IWT with Play Fair Cipher method for encryption. We use an integrated GPS module for motion of the Robot. The process of retrieving the position of the Robot and monitoring its motion has been made simpler by the usage of web-based interface.

Raspberry Pi 3 is used for the motor driven robot. The Pi processes all the inputs based on the information obtained from the GPS module. Planning of the path to be followed is done with the help of an uninformed search technique. The implementation of Bug algorithm detects obstacles efficiently and ensures that repetition of path is avoided which in turn saves time. All information from various sensors is processed at the remote terminal. The Robot moves and detects land mines using magnetometer. A proximity sensor is employed, which helps identifying obstacles in the path.

The main purpose is to detect the landmine and send its location to end-user. As soon as a landmine is detected, the camera module attached to the Robot captures an image of the surroundings. An email is generated through which the captured image is securely transferred to the registered email address. The security of image transmission is guaranteed by the implementation of an encryption mechanism based on DLILL algorithm. The image encryption is made lossless by introducing Integer Wavelet Transform (IWT). The robot is constructed using robust materials to withstand devastating situations like high temperature and pressure. The detection sensor is added to the Raspberry pi 3 and the GPIO pins are assigned to the raspberry pi 3 processor. Whenever the landmine is detected then the pin will turn high and the interrupt will be raised by the raspberry pi 3. Our system uses low cost materials to make the Robot cost effective. Metal detector is used in place of magnetometer which reduces the cost to a great extent.

2. PROBLEM STATEMENT

Most particularly it is very difficult to detect land mines without any risk and casualties. Even though the Government has invented lot of machines and technologies to detect landmines, none of them are completely autonomous. The design of GPS GUIDED ROBOT DRIVEN BY WEB-BASED INTERFACE provides a means to solve this problem.

In order to eliminate the error in the method involving dead reckoning which results in error accumulation, we use the GPS module positioning feature along with an uninformed search technique for the calculation of the position of the robot to proceed further. If we add the dedicated hardware for the GPS then it needs to process the positioning and it should send the coordinates back to the processor. The notable advantage in our proposed system is that it is made to traverse its path autonomously and does not require any explicit control. The identification of location of landmine is made even simpler by the camera module determining the image of the surroundings. The web-based interface clearly depicts the detection of landmines and obstacles along with the exact location. The path planning algorithm determines the direction of the robot to be followed dynamically and sends it to the raspberry pi 3 processor.

By using Bug-0 algorithm memory management is taken care of as state repetition is wisely avoided. Through the built-in Wi-Fi, both the Raspberry Pi 3 and the web-based interface are connected, thus the data transfer is not delayed due to the dedicated connection.

3. LITERATURE SURVEY

Several methods and techniques have been proposed on the detection of landmines and their safe removal. The promising methods used so far include RF bombardment, NQR, ground-penetrating radars, types of neutron energy bombardment, acoustic detection, Infra-red detection, Electric impedance tomography), X-Ray Back scatter and explosive vapor detection [2]. But all have several disadvantages including limitations in sensitivity, technology, outdated hardware, optimal approach and type of metal that can be detected by the landmine detection system. All these are overcome in the model designed in this project.

The mine detecting robot can detect a landmine that is buried up to 5 to 10 meters in depth. It's top speed is 20 cm/sec compared with 15cm/sec in the previous prototype that allows for scanning the area effectively [1]. The path planning algorithm used enables in obstacle deflection of the robot that is an enhancement to the existing prototype helping the robot stay away from detection by the enemy. The robot is faster and uses no external communication module that can be tampered with. The robot can be controlled by remote operator through the wireless medium that is inbuilt. The robot is designed to use the Raspberry Pi-3 which has its processing unit as ARM 1176JZF-S (armv6k) 700MHz and RISC Architecture with low power draw. The performance of R-Pi 3 as compared to ATmega128 is thus vastly improved [1].

The other important aspect is the secured transfer of information regarding the landmines to the appropriate demining personnel to avoid the intervention of antisocial groups. The image encryption using Affine Transform operates on transformation on original image and an XOR operation being performed on this transformed image to get the encrypted image. This system lacks complexity and does not provide complete security. The digital encryption which is based on 1D random scrambling transforms 2D image into 1D vector and then applies random shuffling to perform anti transformation on the shuffled vector to encrypt the image. Unfortunately, this image's histogram reveals excess information so not suitable for highly secretive information. An Alternate technique uses enhanced play-fair cipher along with Integer Wavelet Transform (IWT) to work with the image in spatial and frequency domain and the metrics PSNR, MSE, SSIM and correlation coefficient values makes it a highly secure and efficient method [6].

4. SOFTWARE/HARWARE REQUIREMENT SPECIFICATION

4.1 GENERAL HARDWARE SPECIFICATION OF RASPBERRY PI 3:

a. SYSTEM-ON-CHIP :

The Broadcom BCM2837 chip provides all the features embedded directly on the board instead of building them as individual modules. The cost is greatly reduced owing to this approach.

b. CPU :

It has four high-performance ARM Cortex-A53 processing cores with the speed of 1.2GHz, utilizing 32kB Level 1 and 512kB Level 2 cache memory.

c. GPU :

Broadcom Video Core IV is an integrated Graphic processor that provides Tile Based Rendering (TBR).

d. MEMORY :

1GB LPDDR2 (900 MHz) RAM has been provided which helps in improving the speed of the processor.

e. NETWORKING :

10/100 Ethernet, 2.4GHz 802.11n wireless is the most import feature. The Raspberry Pi 3 controller has in-built wireless connection that can be configured to connect with the Wi-Fi in the location.

f. BLUETOOTH :

Bluetooth 4.1 Classic, Bluetooth Low Energy feature makes it stand unique.

g. GPIO :

40-pin header is used to provide inputs to the Raspbian OS.

h. PORTS :

HDMI, 3.5mm analogue audio-video jack, 4 USB 2.0, Ethernet Serial

Interface (CSI), Display Serial Interface (DSI) have been integrated.

4.2. CONFIGURATION OF RASPBERRY PI 3:

Various commands have been used in the process of configuring Raspberry Pi 3.

a. raspi – config :

Raspberry Pi-3 configuration tool of Raspbian is useful as it helps us to easily use features such as the camera, and alter particular settings such as keyboard layout. Sudo is needed since we will be altering files.

To open the configuration tool, type

```
sudo raspi-config
```

b. config.txt :

A file that is used on the boot partition optionally. The Raspberry Pi file of configuration. It stores the System Configuration Parameters of Raspbian OS. The GPU reads this file before the ARM Core is initialized. It can be accessed using the path,

```
/boot/config.txt
```

c. Device Trees :

They describe the representation of hardware in the system. Allocation of resources, loading and management of modules is done by Device Trees(DT). By this, the problem of managing multiple drivers is eliminated. The hardware configuration is picturized as a hierarchy of nodes. Each node is divided into subnodes and carry different properties.

The Device Tree contains :

A required header : /dts-v1/

A single root node : /

A couple of child nodes : node1 and node2

d. Overlay :

A mechanism used to overcome the limitations of Device Trees. Using overlay, certain fragments can be conditionally included or excluded.

4.3 REQUIRED HARDWARE COMPONENTS:

a. RASPBERRY PI 3:

A Single Board Computer with 64 bit processor that is ARM based and supports on-board Wi-Fi is the base for processing data in this project.

b. IR SENSORS:

They are based on the concept of reflection of IR signals when they hit a surface. The IR Receiver receives the signals that bounce from the obstacle.

c. L293D:

A 16-pin Integrated Circuit that is capable of controlling the two motors used for movement of the Robot. It directs the motors simultaneously.

d. PROXIMITY SENSOR:

It emits electromagnetic signals and detects the presence of obstacles by observing changes in the signal that is returned.

e. MOTORS:

These are the actuators that act in response to the commands from the web-based interface and the Pi 3 processor. The motors maintain constant torque in order to enable movement in different directions.

f. MAGNETOMETER:

The magnetometer HMC1022 is proposed to be used here as well. It's field ranges reach up to ± 6 Gauss. It also enables tri-axis representation for sensitivity depiction. Additionally, its internal components eliminate effects due to signal loss and variation in temperature.

g. CHASIS & TYRES:

The chassis provides the base for accommodating all the constituting components of the Robot and the tires provides means of movement.

h. JUMPER WIRES:

There are the wires that connect the pins of the Pi 3 processor to the motors and the motor driver.

4.4 SOFTWARE REQUIREMENTS SPECIFICATION:

Debian's Apache2 Server Installation and Backend Connection to web application

The software requirements mainly involve connecting the web application to the robot so that input can be given to the device. Thus the Raspberry Pi module can get inputs from the server from the web application and give appropriate commands to its components. Debian's Apache2 server is mainly installed on Raspbian OS in the Raspberry Pi 3. It is a light-weight server which can respond to client requests very quickly. For this project the client is the remote operator operating the web application and the server is Raspberry Pi 3.

For the Raspbian OS, apache2 server is installed using the following command on command prompt. The command is:

sudo apt-get install apache2 -y

Through this command, first apache2 package gets installed. Once this installation is done then web server must be tested. To do this, <http://localhost/> can be visited on PI. This default webpage is located at a directory “**/var/www/html**” on Raspbian file system. After web server is tested, then PHP module is installed for Apache. This is achieved through following command

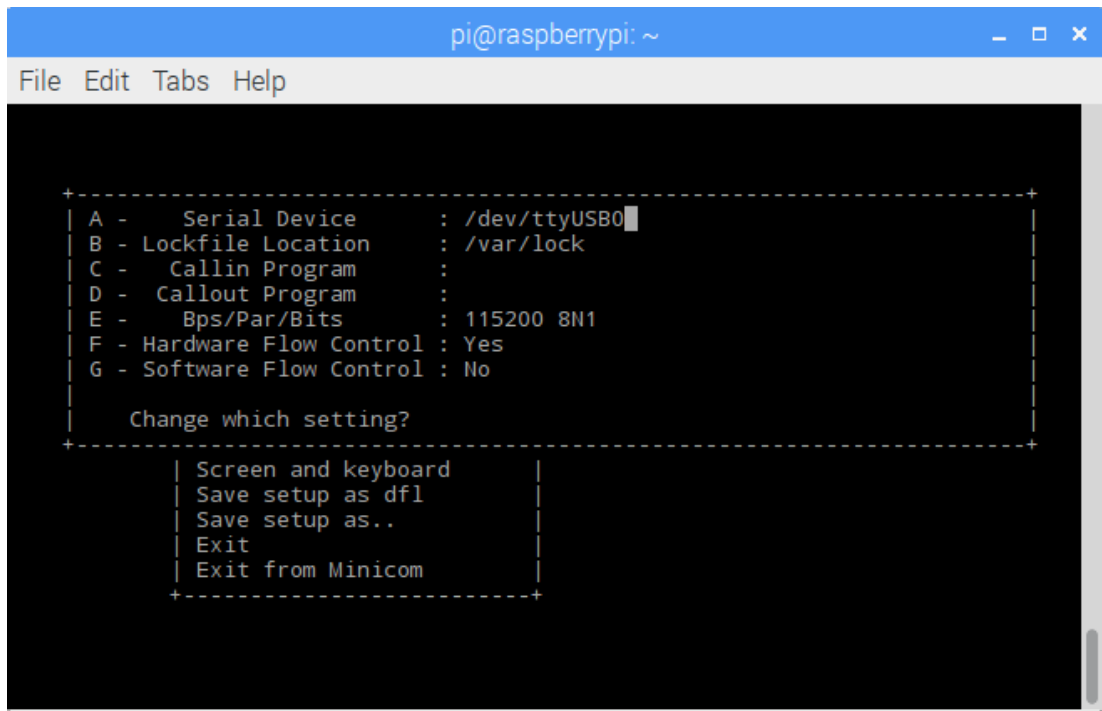
sudo apt-get install php5 libapache2-mod-php5 -y

Now, PHP files can be processed by Apache server. An index php file is created and run on Apache server. This index php file is created through following command

sudo nano project.php

This project.php file is accessed through IP address of the raspberry pi.

.

A screenshot of a terminal window titled 'pi@raspberrypi: ~'. The window contains a menu for configuring a serial connection using Minicom. The menu is enclosed in a dashed border and lists options A through G: A - Serial Device : /dev/ttyUSB0, B - Lockfile Location : /var/lock, C - Callin Program :, D - Callout Program :, E - Bps/Par/Bits : 115200 8N1, F - Hardware Flow Control : Yes, and G - Software Flow Control : No. Below this list, it asks 'Change which setting?'. At the bottom, another dashed box contains options: Screen and keyboard, Save setup as dfl, Save setup as.., Exit, and Exit from Minicom.

```
pi@raspberrypi: ~
File Edit Tabs Help

+-----+
| A -   Serial Device       : /dev/ttyUSB0 |
| B - Lockfile Location    : /var/lock     |
| C -   Callin Program      :               |
| D -   Callout Program     :               |
| E -   Bps/Par/Bits        : 115200 8N1    |
| F - Hardware Flow Control : Yes           |
| G - Software Flow Control : No           |
+-----+
      Change which setting?
+-----+
| Screen and keyboard      |
| Save setup as dfl       |
| Save setup as..         |
| Exit                    |
| Exit from Minicom       |
+-----+
```

Fig 1: Command prompt of Raspberry Pi OS

Fig 1 shows a screenshot of the command prompt screen for Raspbian OS.

The screenshot shows the various settings that can be changed using the command prompt. Some of those shown include Serial Device settings, Location, Software and Hardware Flow Control settings etc. We can then customize each setting and name it as our own. Similarly, keyboard and screen default settings can be changed. Thus it gives users flexibility to use it the way they want to.

5. CONCEPTUAL MODEL / PROPOSED ARCHITECTURE

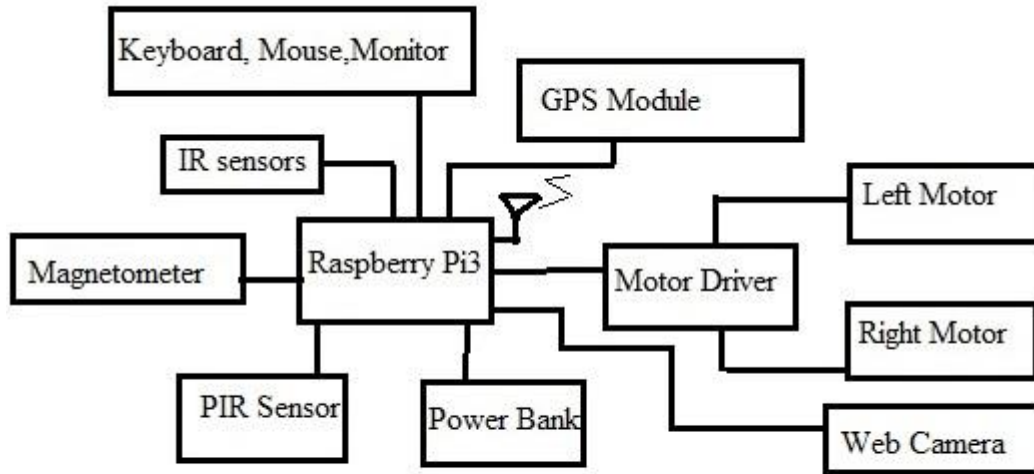


Fig 2: Basic block diagram of the proposed system

The prototype design is depicted in Fig 2. The magnetometer connected to the Raspberry Pi3 is used for sensing the landmines. The update from AT Mega 128 microcontroller to Raspberry Pi3 has enabled better and easier way of interfacing with mine detection sensors. The localization data from the GPS module is dynamically conveyed to the remote terminal that greatly reduces memory requirement from Raspberry Pi3's point of view. The UI developed for the control of the robot reports status of detection of metals or obstacles based on sensor information.

Differential steering along with the GPS is used in marking the correct location of the bot. A high precision proximity sensor and two other IR sensors position at the right and left side of the robot helps in obstacle detection and corresponding deflection based on bug algorithms.

The in-built Wi-Fi in Pi 3 helps establish communication between the Pi in the robot and remote control stations. An uninformed search technique helps the robot to traverse it from a given location to another without the actual knowledge of the entire environment. The robot called Mine Hunter Beta version has many physical and economic highpoints such as area dimensions of 18.5 cm \times 9.5 cm and weighs only 0.77 kg. The usage of two 9V batteries to power the wheels, power source for motion can be replaced in quite an economic way. Mine Hunter Beta Version has an on board rechargeable power bank that can power it for 4 hours, that is significantly more than the previous prototype's power time.

GPIO PINS FOR RASPBERRY PI 3:

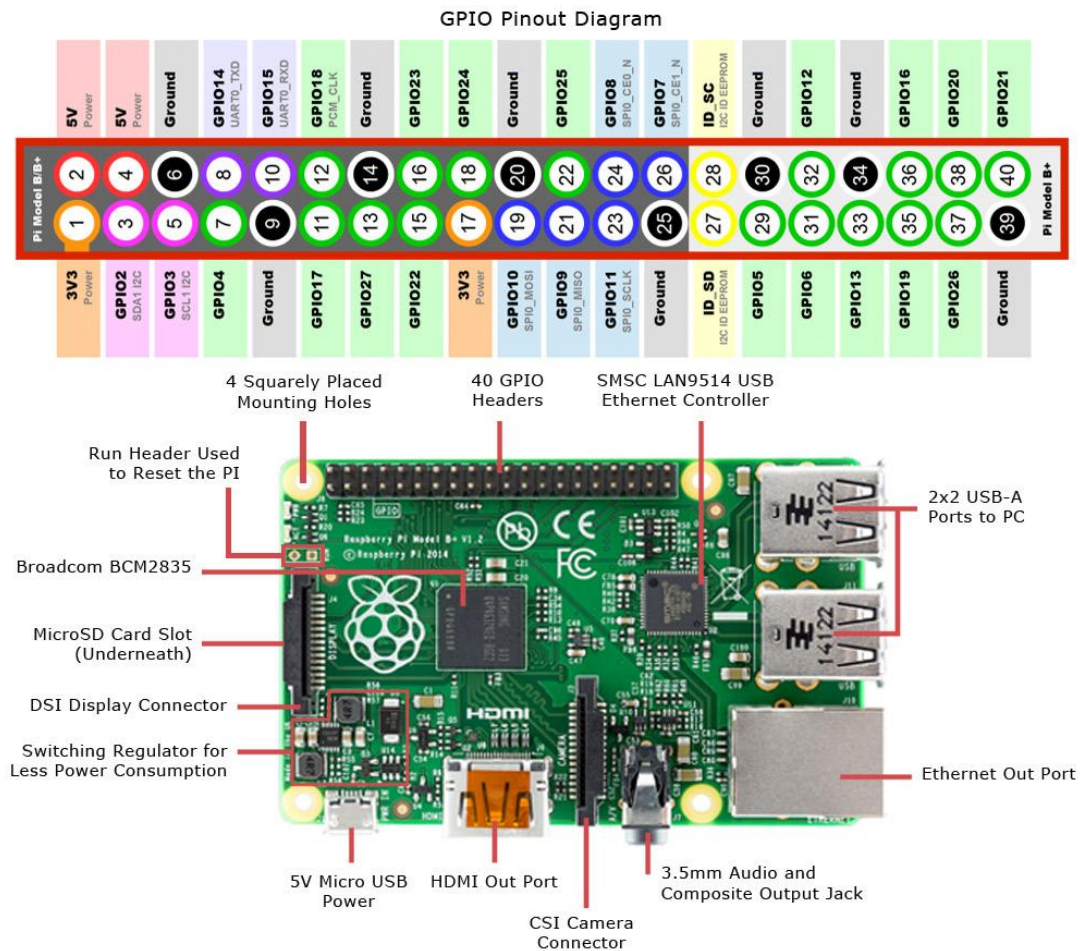


Fig 3: GPIO Pin Diagram for Landmine Detection Robot controlled web-based user interface

Fig 3 shows the GPIO pins. GPIO pins can be used as both digital outputs and digital inputs.

Output: turn a particular pin HIGH or LOW.

- Setting it HIGH sets it to 3.3V;
- Setting it LOW sets it to 0V.

Input: detect the pin being at HIGH or LOW

We can connect switches and simple sensors to a pin and check whether it is open or closed (that is, activated or not)

COMPONENTS OF THE SYSTEM:



FIG - 4 : Components used

Figures 4 show the various components of the robot. The main sensors used here are the PIR sensor and the magnetometer. Whereas the PIR sensor is used for detecting obstacles in the front side of the robot, the Magnetometer is used for detecting landmines buried upto 5 to 10 meters in depth. Along with the PIR sensors, two IR sensors attached to either sides of the robot help in detecting obstacle boundaries.

VARIOUS VIEWS OF THE ROBOT :

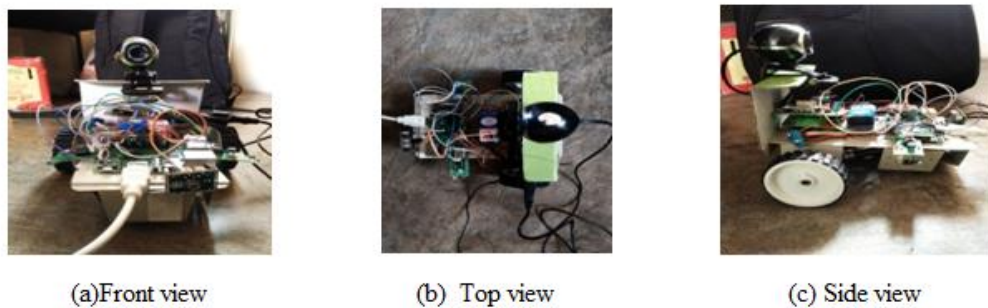


FIG – 5 : Various views of the Robot

The various views of the bot are represented in Fig 5. The camera is inserted at the top attached to the bot with the help of a vertical slab. The proximity sensor and magnetometer are present in the front side. In the side view an IR sensor is seen. Another similar sensor is placed at the other side as well to detect obstacles on both sides of the bot and help in deflection.

6. INTERACTION SCENARIO

6.1 USE CASE DIAGRAM:

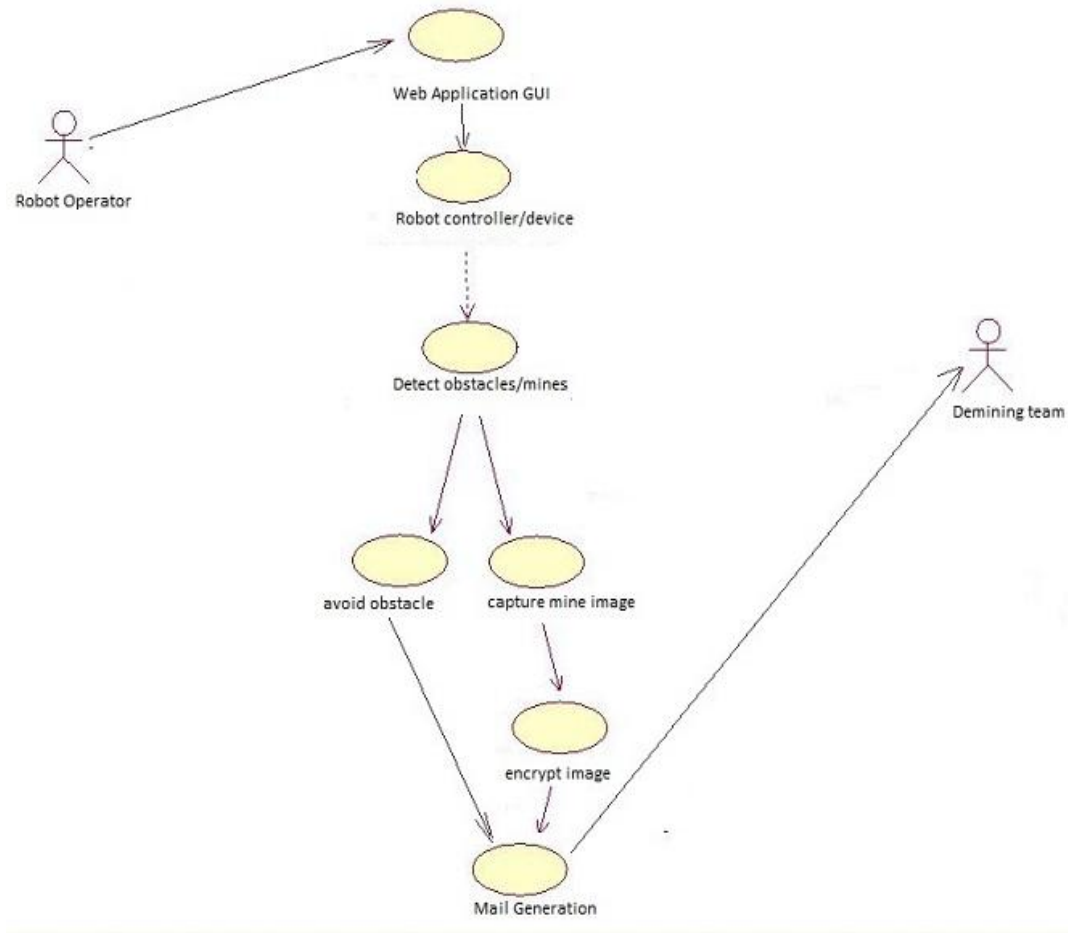


Fig 6: Use case diagram for landmine detection robot

The use case diagram (Fig: 6) process flow is as follows:

- 1. Robot operator to web application interface:** In this Use case diagram shown in Fig 6, end-user is the remote controller who is interacting with the robot for its operation. The controller interacts with the robot through web application which acts as an interface to Debian's Apache2 server php program. The input given in terms of the button pressed (be it start or stop button) is processed through PHP program in Apache2 Server.
- 2. Operator gives input to device through server:** On processing this data in server, these values and inputs will be passed to Raspberry Pi 3 processor, which connects to L293D that drives the motor, moves along a path with the help of uniformed search technique, and detect landmines.
- 3. The robot moves along the path obtaining sensor information:** Along the way if obstacles are encountered then path is changed and obstacle is

deflected or if landmines are detected then an image of the surroundings is captured through camera module and sent along in an encrypted format using IWT and Play-Fair Cipher method to the demining team.

6.2 ACTIVITY DIAGRAM

6.2.1 ACTIVITY DIAGRAM FOR WEB APPLICATION TO APACHE2 SERVER

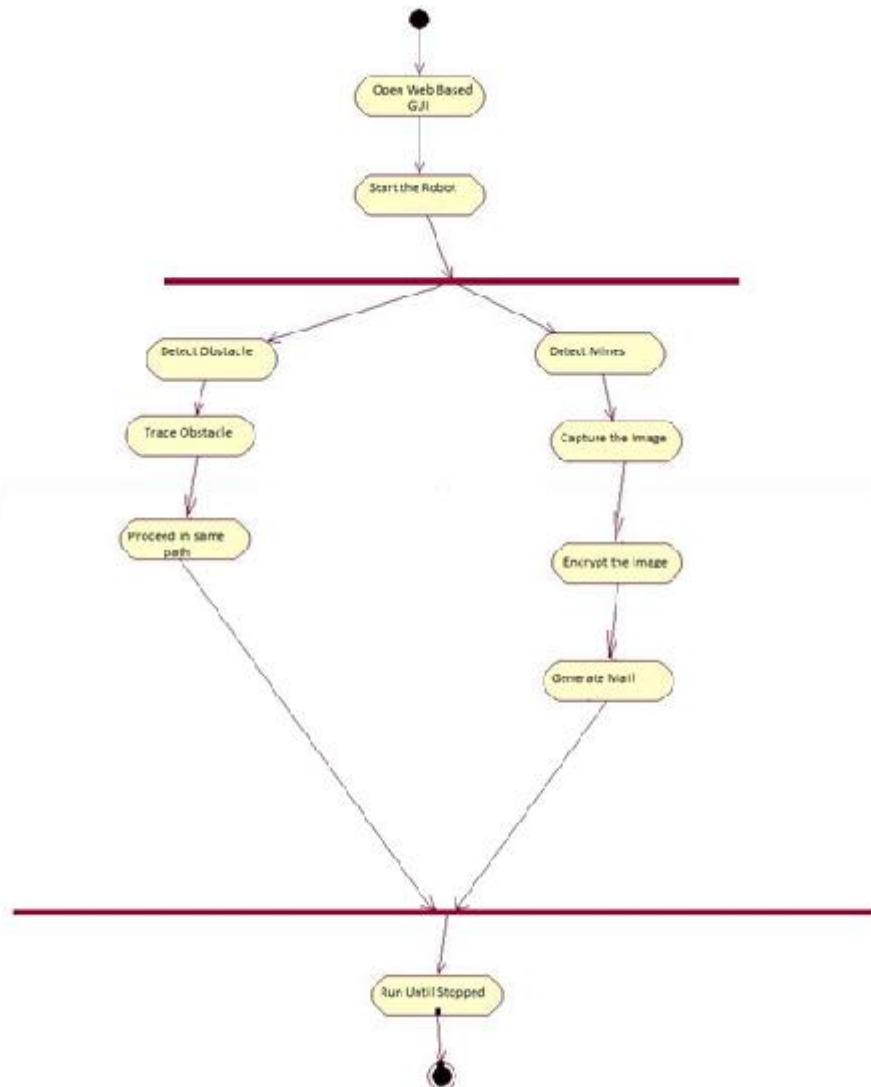


Fig 7(a) Activity diagram for landmine detection robot

The activity diagram (Fig 7(a)) process flow is as follows:

- 1. Robot operator to web application:** Via the connection between the web application and Debian's Apache2 server inputs received at the GUI are sent to the Raspberry Pi 3 processor. The web application provides an interface to the PHP program run on the server. This in turn connects to the L293D module which drives Mine Hunter Beta version's motor and moves on the area
- 2. Selecting start or stop Button:** Data in the PHP Server will be passed as

inputs to the Raspberry Pi 3 processor. If start is pressed the robot will be in motion and when stop is pressed it comes to a halt.

3. **It deflects obstacles and detects landmines:** Two of its main functions are these two. While in motion sensory information is received and continuously relayed to the processor.
4. **On landmine detection mail is sent with encrypted information:** Sending valuable information to the demining team regarding the landmines location in an encrypted format.
5. **If stop is pressed, robot stops its movement.**

6.2.2 ACTIVITY FLOW DIAGRAM FOR MOTOR MOVEMENT

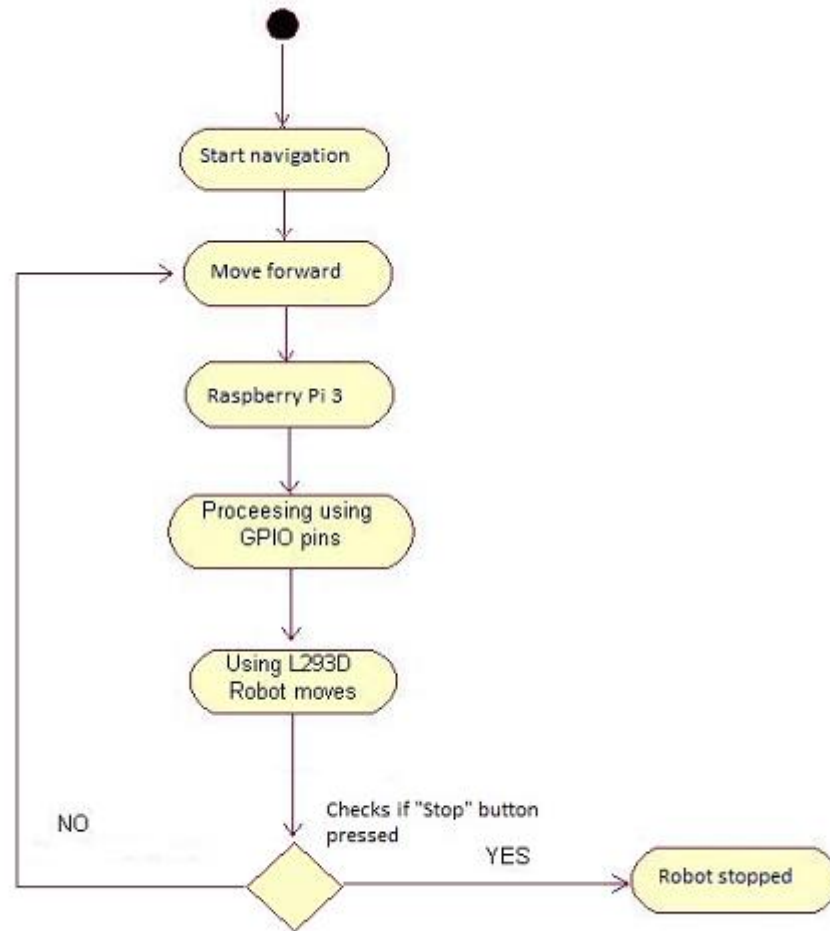


Fig 7 (b): Activity diagram for motor movement in landmine detection robot

The activity diagram (Fig 7 (b)) process flow is as follows:

1. **PHP server input data to Raspberry Pi-3 processor:** Based on button pressed in PHP server, the input is passed to processor which in turn connects to L293D that drives the motor and moves along the path.
2. **Raspberry Pi-3 processor to GPIO pins in it:** Deflecting obstacles on the way and detecting landmines whenever magnetometer and other sensors send information to the processor from the GPIO pins.
3. **GPIO pins to motor using L293D:** These pins provide information to actuators to perform the required actions. Thus the GPIO pins are extremely important with regards to robot movement.

6.3 CLASS DIAGRAM

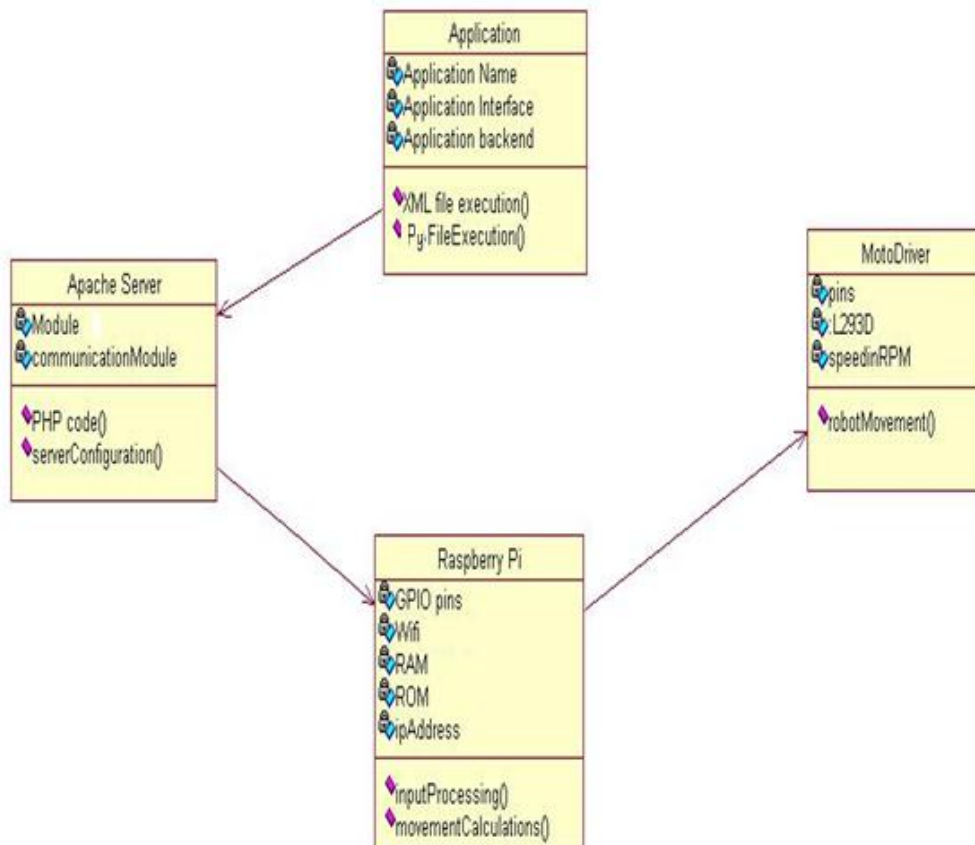


Fig 8: Class Diagram for landmine detection robot

The class diagram shown above in Fig 8 has four classes namely, Application, Apache Server, Raspberry Pi and Motor Driver. These 4 classes consists of the main components involved in this project thus in turn helps in achieving project objective. Each of these classes contains many attributes that decides the properties of class. They interact with each other for successful functioning of bot.

1. **Application class:**

It consists of members such as application front end containing the XML, the Application name, the interface itself and the backend containing the python code. The codes are the functional component of the project and thus play a major role in its output.

2. **Apache Server:**

The Debian's Apache2 server is important for communication between two major classes namely the application and Raspberry Pi-3 processor. The communication module is to establish contact between the two. The PHP code is the medium through which it is established. The server configuration is set to default.

3. **Raspberry Pi:** This class contains many members like GPIO pins, RAM, ROM, Wi-Fi, IP address etc. Each of these is used in the project and plays an important role in the coming together of the system as a whole functioning system.
 - a) `inputProcessing()` function is used for receiving inputs from the Server and relaying it to the Motor Driver.
 - b) Before doing so the input is converted in to a form that is understood by the Motor Driver using `motorCalculation()` function.
4. **Motor Driver:** The motor driver receives inputs from the Pi using L293D and does the robot movement using the function present in it. The sensor information is also required to determine the kind of movement performed for example if obstacle is found the path is changed until obstacle is no longer present.

6.4 SEQUENCE DIAGRAM

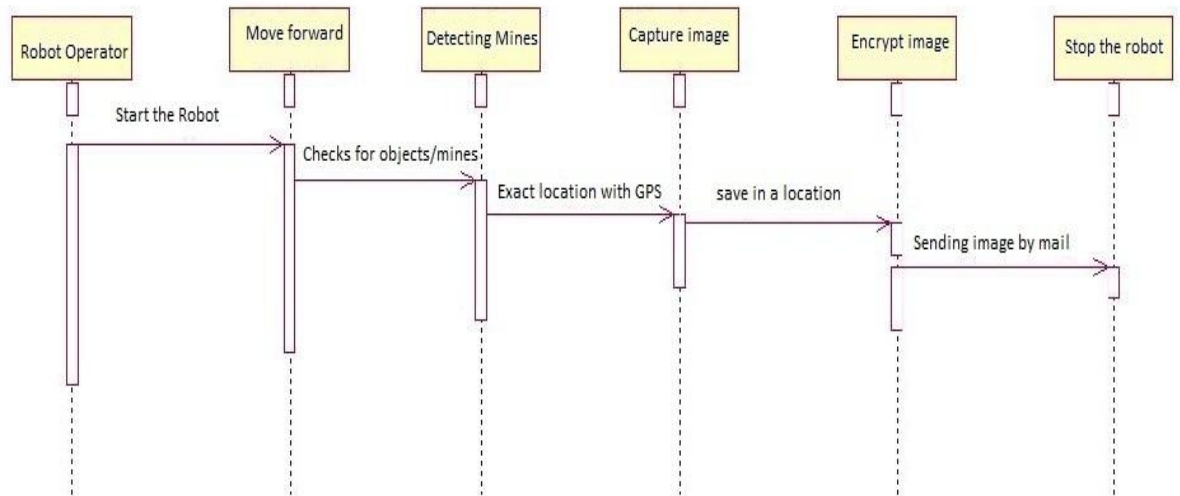


Fig 9: Sequence Diagram for landmine detection robot

The sequence diagram (Fig.9) process flow is as follows:

- 1. Remote operator to web application interface:** At each stage the input received from the web application is sent to the Raspberry Pi 3 processor through the Apache 2 server. As the input start button is pressed the Raspberry pi 3 actuators give input to motors to start moving.
- 2. This processed data is sent to Raspberry pi and the robot moves by uninformed search technique:** According to the path planning technique used the robot moves along a path at the same time looking for obstacles or landmines. If an obstacle is encountered the robot deflects from the obstacle and follows the boundary of the obstacle until it can move along the same path again.
- 3. Robot detects landmines:** Once a landmine is detected the camera module in the robot captures an image and sends it to the demining team mail id.
- 4. Robot sends Encrypted mail on landmine detection:** To ensure more secure transfer of the landmine location it encrypts the image and sends it. All this while the GPS module keeps track of robot location

6.5 COLLABORATION DIAGRAM

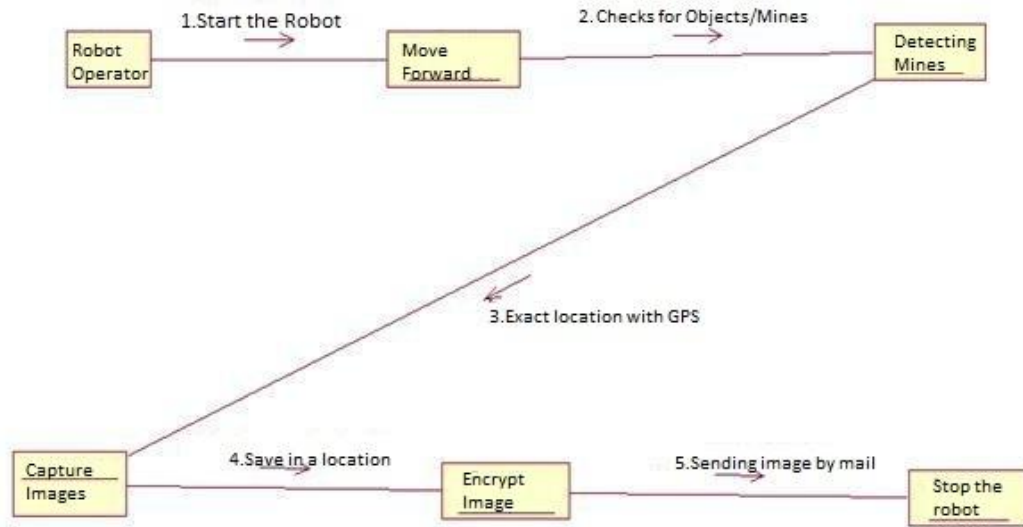


Fig 10: Collaboration diagram for landmine detection robot

This shows the communication between the objects which transfer information through send and receive messages.

The collaboration diagram (Fig 10) process flow is as follows:

- 1. Robot operator uses the web-application interface**
- 2. Web application interface gives status:** It depicts the dynamic nature of this system. Various objects like Robot Operator, Apache sever communicate with each other and transfer system related messages that helps in its success.
- 3. Raspberry Pi-3 traverses path with uninformed search:** Uses Bug algorithm for path planning and obstacle detection
- 4. Landmines are detected by giving status info and command line:** Based on sensory information the detection is done and location of landmine is calculated using GPS module.
- 5. Image is captured, encrypted and sent through mail:** With the code for encryption the image captured is secured and sent via mail.

7. METHODOLOGY AND APPROACH

Module A: Assembling Robot and Integrating with Remote Desktop

The Robot is assembled with all its components controlled by Raspberry Pi 3 micro-controller, operated through the remote desktop connection with a client server connection through the static IP of the raspberry pi. Raspberry pi consists of a memory interface where Raspbian operating system is loaded and used for accessing the files that are loaded into the Micro-controller. Input is given from the connected desktop to controller board, which sends the necessary signal to the actuators (motors) depending on the input received. Establishing remote desktop connection is done by the built-in WiFi, a powerful feature incorporated in Raspberry Pi 3. The Raspbian operating system is loaded into the desktop for accessing, executing the tasks of the robot.

The following code acts as the PHP code and interface to the robotic operator.

PHP CODE:

```
<html>
<head>
<meta name="viewport" content="width=device-width" />
<title>Robot Control</title>
</head>
  <body>
    WEB PAGE ON PHP BASED Robot Control:
    <form method="get" action="emb.php">
      <input type="submit" value="START" name="Start">
      <input type="submit" value="STOP" name="Stop">
    </form>
    <?php
    $setmode26 = shell_exec("/usr/local/bin/gpio -g mode 26 out");
    if(isset($_GET['start'])){
      $anbuemb_on = shell_exec("/usr/local/bin/gpio -g write 26 1");
      echo "ROBOT is start";
    }
    else if(isset($_GET['stop'])){
      $anbuemb_off = shell_exec("/usr/local/bin/gpio -g write 26 0");
      echo "ROBOT is stop";
    }
    ?>
  </body>
</html>
```

```
<?php
$myfile = fopen("/home/pi/chennai/log.txt", "r") or die("Unable to open file!");
echo fread($myfile,filesize("/home/pi/chennai/log.txt"));
fclose($myfile);
?>
```

The start and stop button in the PHP code is linked to the processor via the Apache2 server. The input is thus given to Pi 3 which then relays the message to its actuators.

Apache2 Server:

Apache Server is a light-weight server which can respond to client requests very quickly. In this project, Debian's Apache2 default configuration is used which is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools.

Module B: Integrating web-based interface with the Robot to obtain the GPS location and drive the Robot depending on the path planning algorithm.

At this stage of project, we interface the robot with a web-based application deployed in a desktop. The web application use the path planning algorithm, based on bug algorithm and an uninformed search technique for navigation. The application is guided by a GPS module to denote the location of the robot and the obstacles frequently. Based on the decisions obtained from the algorithm, the subsequent direction of motion of the robot is identified and the robot is made to trace through the path for detecting the landmines.

Interfacing with Web application

The PHP code that is shown in our web interface controls the robot. After interfacing the PHP code to our processor we execute the back-end python code on the click of the start button on the interface. The controller can stop the robot anytime he wants by the stop button. On each button click the following python code is performed.

Python Module 1:

```
while True:
```

```
    if(GPIO.input(13) == 1):
        print('metal Detected')
        variab='metal Detected'
```

```

f=open('log.txt','w')
f.write(variab)
f.close()
time.sleep(1)
print('CAPTURING IMAGE')
GPIO.output(27, True)
subprocess.Popen( "fswebcam -r 1280x720
                    /home/pi/chennai/download.png", shell=True )
    time.sleep(1)
    subprocess.Popen("python /home/pi/chennai/mail.py", shell=True)
    time.sleep(1)
else:
    print ('Nearby no metals present')
    time.sleep(1)
    variab=' Nearby no metals present'
    f=open('log.txt','w')
    f.write(variab)
    f.close()
    time.sleep(1)

if(GPIO.input(6) == 1):
    print('Obstacle Detected')
    variab='Obstacle detected'
    f=open('log.txt','w')
    f.write(variab)
    f.close()
    time.sleep(1)
    print('CAPTURING IMAGE')
    GPIO.output(27, True)
    subprocess.Popen( "fswebcam -r 1280x720
                        /home/pi/chennai/download.png", shell=True )
        time.sleep(1)
        subprocess.Popen("python /home/pi/chennai/mail.py", shell=True)
        time.sleep(1)

```

In the above python code module, the pins are interfaced as input or output pins specific numbers assigned to them. Packages are included to call one program within another (for mail segment) and also for GPS module explained later. The pins are physically connected to sensor components of the bot. The uninformed search technique namely Bug algorithm helps the robot traverse a particular path and send and relay messages regarding landmine location (through the GPS module) to remote locations either via statuses or mails sent. Meanwhile, on the event of output from a sensor the corresponding check is being done and the status displayed from the file 'log.txt'.

Python mail module:

```
msg = MIMEMultipart()
time.sleep(1)
msg['Subject'] = "testing msg send from python"
time.sleep(1)
fp = open("/home/pi/chennai/endownload.png", 'rb')
time.sleep(1)
img = MIMEImage(fp.read())
time.sleep(1)
fp.close()
time.sleep(1)
msg.attach(img)
time.sleep(1)
    server.login(gmail_user, gmail_pwd)
        print "reading mail & password"
    server.sendmail(FROM, TO, msg.as_string())
        print "from"
    server.close()
    print 'successfully sent the mail'
```

This module is called on detection of the landmines. Required packages such as smtplib and base64 are downloaded using sudo commands. The sender and receiver mail ids are predefined and server is being initialized to the mail address. On improper signal from the router the mail program waits for a maximum of 10 seconds when it displays an error in mail transfer and continues execution of the main program. On successful wireless connection however, this mail module then calls the image encryption module that uses a Play-Fair Cipher technique to encrypt image containing landmine location and sends the encrypted image.

Path planning module using Bug Algorithm with GPS module implementation

Using Bug algorithm, the Robot can locally sense obstacles and move along the obstacles until it finds a path. Along with an uninformed search technique that has been incorporated, the path planning can be made spontaneous and accurate to a great extent.

```
while True:
    rcv = port.read(1)
    while(rcv is not 'L'):
        rcv = port.read(1)
    if(rcv=='L'):
        for i in range(0,25):
```



```

rcv = port.read(1)
data = data+rcv
variab='L13.539,,192132.086,V,N*7E'+data
f=open('log.txt','w')
f.write(vариab)
f.close()

print data
if(GPIO.input(13) == 1):
    GPIO.output(27, True)
    print("metal detected at gps value:"+data)
    variab='metal Detected'
    f=open('log.txt','w')
    f.write(vариab)
    f.close()
    time.sleep(1)
    print('CAPTURING IMAGE')

    subprocess.Popen( "fswebcam -r 1280x720
                        /home/pi/chennai/download.png", shell=True )
    time.sleep(1)
    subprocess.Popen("python /home/pi/chennai/mail.py", shell=True)
    time.sleep(1)

```

The additional snippet added to the basic code shows that Bug algorithm is implemented along with a GPS module locating the robot every 10 seconds or so.

Module C: Sending Mail to the End-User upon detection of Landmines.

In this phase of project, we attach a landmine detection module (metal detector IC) to the robot. In the location where landmine gets detected an interrupt is generated and sent to Raspberry Pi 3 controller which in-turn sends a signal to the web interface. The web-based interface receives this signal from raspberry pi and sends corresponding GPS location and an encrypted image of the landmine's surroundings to the registered user through mail.

A separate file is prepared that contains the DLILL algorithm implementation. This code then has to be called every time on the detection of landmine-like objects through sensor information.

Python code for Secure Image Encryption and Decryption :

DLILL Encryption :

```
while(m<len(in_image)):
    x = in_image[m]
    y = in_image[m+1]
    for i in range(16):
        for j in range(16):
            if playkey[i][j] == x:
                ix = i
                jx = j
            if playkey[i][j] == y:
                iy = i
                jy = j
        if x == y:
            if ix == 15:
                ix = -1
            if jx == 15:
                jx = -1
            cipher_image.append(playkey[ix+1][jx+1])
            cipher_image.append(playkey[ix+1][jx+1])
        elif ix == iy and jx != jy:
            if jx == 15:
                jx = -1
            if jy == 15:
                jy = -1
            cipher_image.append(playkey[ix][jx+1])
            cipher_image.append(playkey[iy][jy+1])
        elif ix != iy and jx == jy:
            if ix == 15:
                ix = -1
            if iy == 15:
                iy = -1
            cipher_image.append(playkey[ix+1][jx])
            cipher_image.append(playkey[iy+1][jy])
        elif ix != iy and jx != jy:
            cipher_image.append(playkey[ix][jy])
            cipher_image.append(playkey[iy][jx])
        m += 2
    final_image = []
    c = 0
    for j in range(dim[1]):
```

```

final_image.append([])
for i in range(dim[0]):
    final_image[j].append(cipher_image[c])
    c += 1

```

Using DLILL algorithm, in the image captured by the camera module, the IWT (Inverse Wavelet Transform) is applied to get 4 different matrices with transformed values. Leaving the LL matrix undisturbed while encrypting all the 4 matrices obtained by IWT, followed by combining all the matrices and applying Inverse IWT gives the resultant encrypted image. This method gives a maximum encryption with minimal loss in the image data.

DLILL Decryption :

```

while(m<len(in_image)):
    x = in_image[m]
    y = in_image[m+1]
    for i in range(16):
        for j in range(16):
            if playkey[i][j] == x:
                ix = i
                jx = j
            if playkey[i][j] == y:
                iy = i
                jy = j
        if x == y:
            if ix == 0:
                ix = 16
            if jx == 0:
                jx = 16
            cipher_image.append(playkey[ix-1][jx-1])
            cipher_image.append(playkey[ix-1][jx-1])
        elif ix == iy and jx != jy:
            if jx == 0:
                jx = 16
            if jy == 0:
                jy = 16
            cipher_image.append(playkey[ix][jx-1])
            cipher_image.append(playkey[iy][jy-1])
        elif ix != iy and jx == jy:
            if ix == 0:
                ix = 16
            if iy == 0:

```

```

        iy = 16
        cipher_image.append(playkey[ix-1][jx])
        cipher_image.append(playkey[iy-1][jy])
    elif ix != iy and jx != jy:
        cipher_image.append(playkey[ix][jy])
        cipher_image.append(playkey[iy][jx])
    m += 2
final_image = []
c = 0
for j in range(dim[1]):
    final_image.append([])
    for i in range(dim[0]):
        final_image[j].append(cipher_image[c])
        c += 1

```

The advantages of this technique is that unlike all other image encryptions which works either in spatial or frequency domains, this method combines both the frequency domain (by conversion from spatial to frequency domain) and spatial domain (after encryption of frequency domain intermediate) aspects on decryption of the image there is zero mean square error and the original image is obtained back lossless.

8. OUTPUT/RESULTS

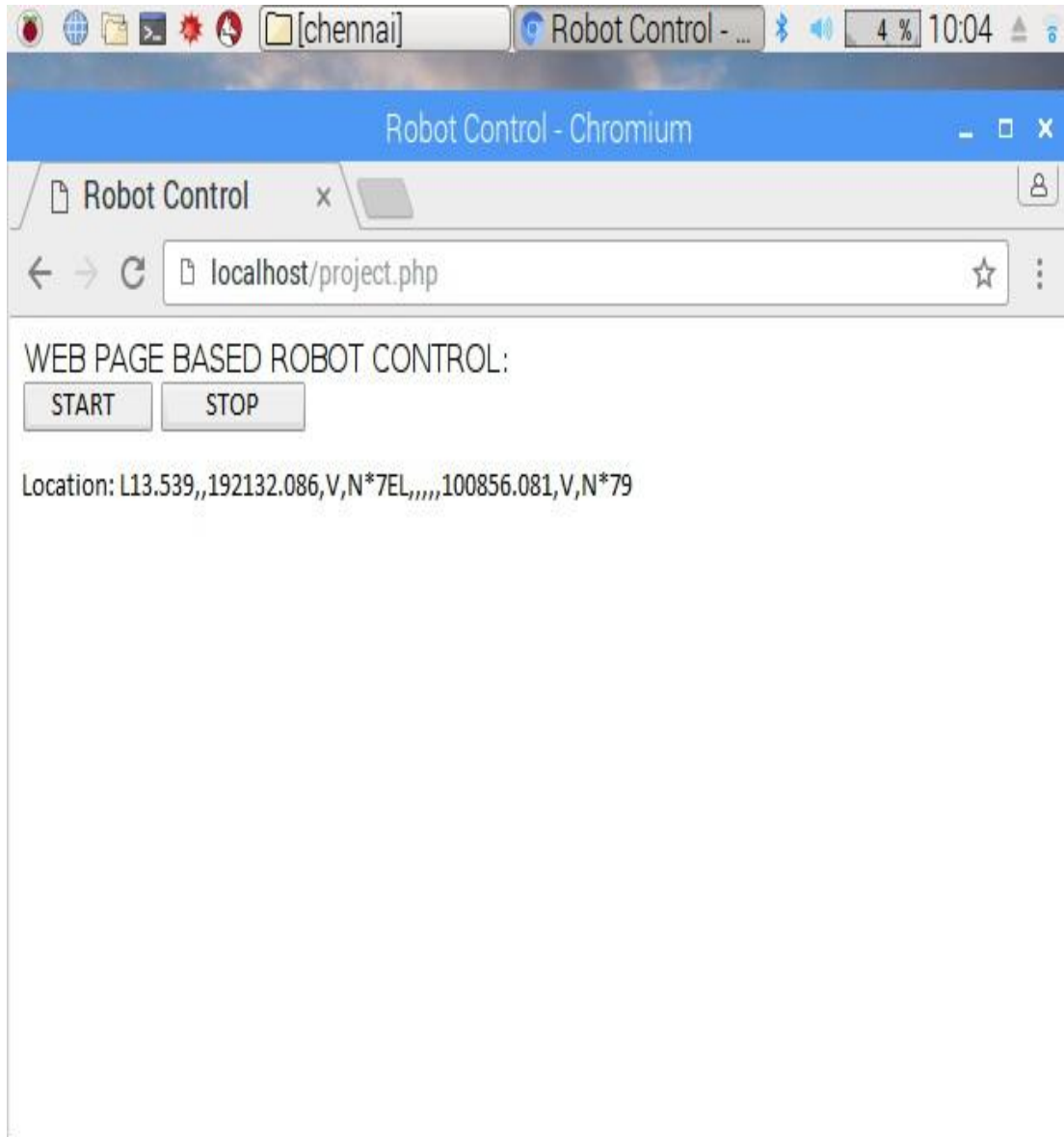


Fig 11: Output Screen of Web application controlling Robot

In the above Fig 11, the web application is shown that can help remote users control the robot. Since we have designed it for an autonomous operation it contains only start and stop buttons. On start the robot starts its motion. Anytime the remote operator needs to stop the robot it can be. Due to this it can be both autonomous and in control of the operator.

The various cryptographic algorithms run on Raspberry pi 3 shows the following results as in FIG 12.

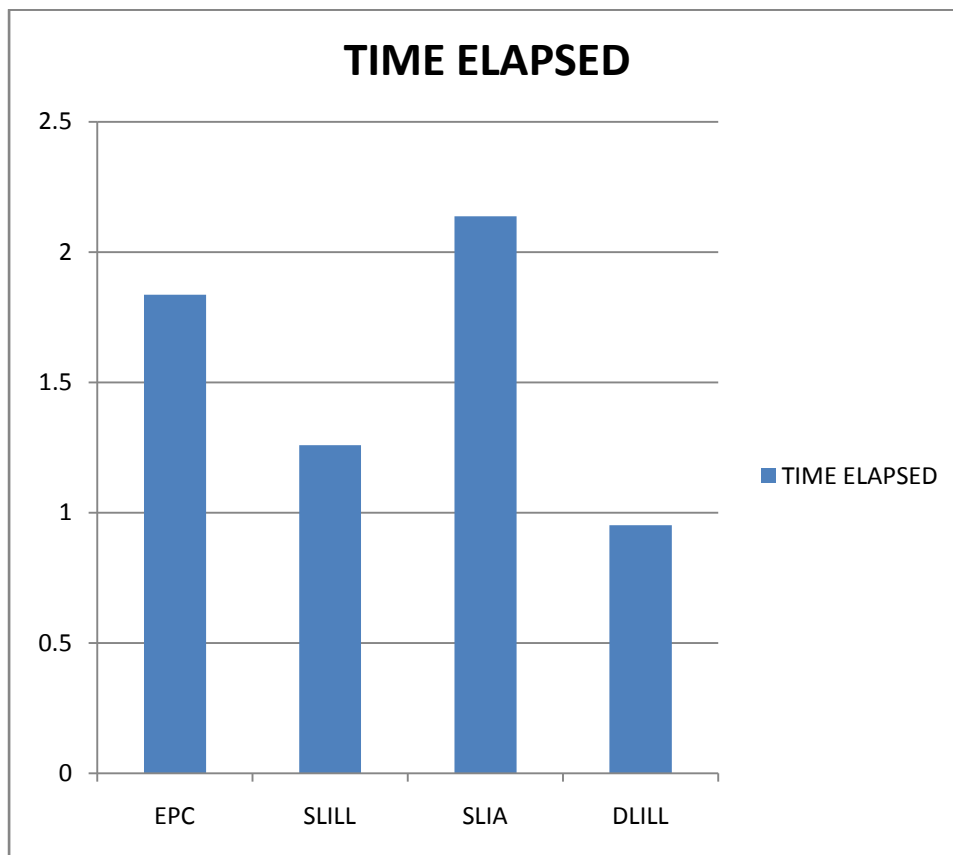
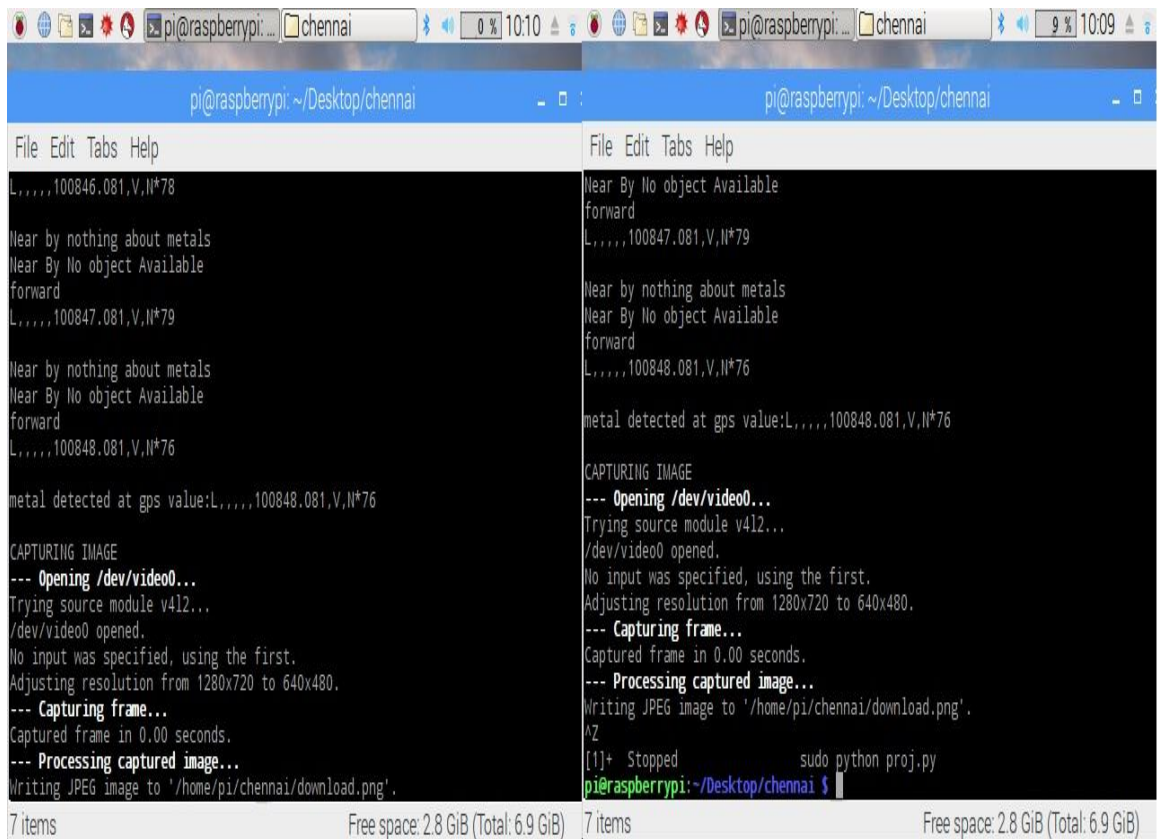


FIG 12- Method versus Time elapsed for various algorithms

Among the various methods considered for encryption the DLILL is a novel and innovative way to approach image encryption. Moreover the technique is also very optimal. DLILL method for encryption implemented in the prototype shows lesser response time.



```
pi@raspberrypi: ~/Desktop/chennai
File Edit Tabs Help
L,,,,,100846.081,V,N*78
Near by nothing about metals
Near By No object Available
forward
L,,,,,100847.081,V,N*79
Near by nothing about metals
Near By No object Available
forward
L,,,,,100848.081,V,N*76
metal detected at gps value:L,,,,,100848.081,V,N*76
CAPTURING IMAGE
--- Opening /dev/video0...
Trying source module v4l2...
/dev/video0 opened.
No input was specified, using the first.
Adjusting resolution from 1280x720 to 640x480.
--- Capturing frame...
Captured frame in 0.00 seconds.
--- Processing captured image...
Writing JPEG image to '/home/pi/chennai/download.png'.
[1]+ Stopped sudo python proj.py
pi@raspberrypi:~/Desktop/chennai $

pi@raspberrypi: ~/Desktop/chennai
File Edit Tabs Help
Near By No object Available
forward
L,,,,,100847.081,V,N*79
Near by nothing about metals
Near By No object Available
forward
L,,,,,100848.081,V,N*76
metal detected at gps value:L,,,,,100848.081,V,N*76
CAPTURING IMAGE
--- Opening /dev/video0...
Trying source module v4l2...
/dev/video0 opened.
No input was specified, using the first.
Adjusting resolution from 1280x720 to 640x480.
--- Capturing frame...
Captured frame in 0.00 seconds.
--- Processing captured image...
Writing JPEG image to '/home/pi/chennai/download.png'.
[1]+ Stopped sudo python proj.py
pi@raspberrypi:~/Desktop/chennai $

7 items Free space: 2.8 GiB (Total: 6.9 GiB) 7 items Free space: 2.8 GiB (Total: 6.9 GiB)
```

Fig 13: Output Screen showing snippets of the main code execution

In the above output screen (Fig 13), a sample execution of the program is shown. Note GPS locations shown in the program that would inform the remote operator of the mine location. On successful detection a mail is also sent that is encrypted. The status shown in this command prompt can be viewed in the interface of the web application as well. This is done by writing all sensor outputs in a separate log.txt file and updating it on the PHP page.

Figures 14 shows the mail generated from one mail id to another with the image sent.

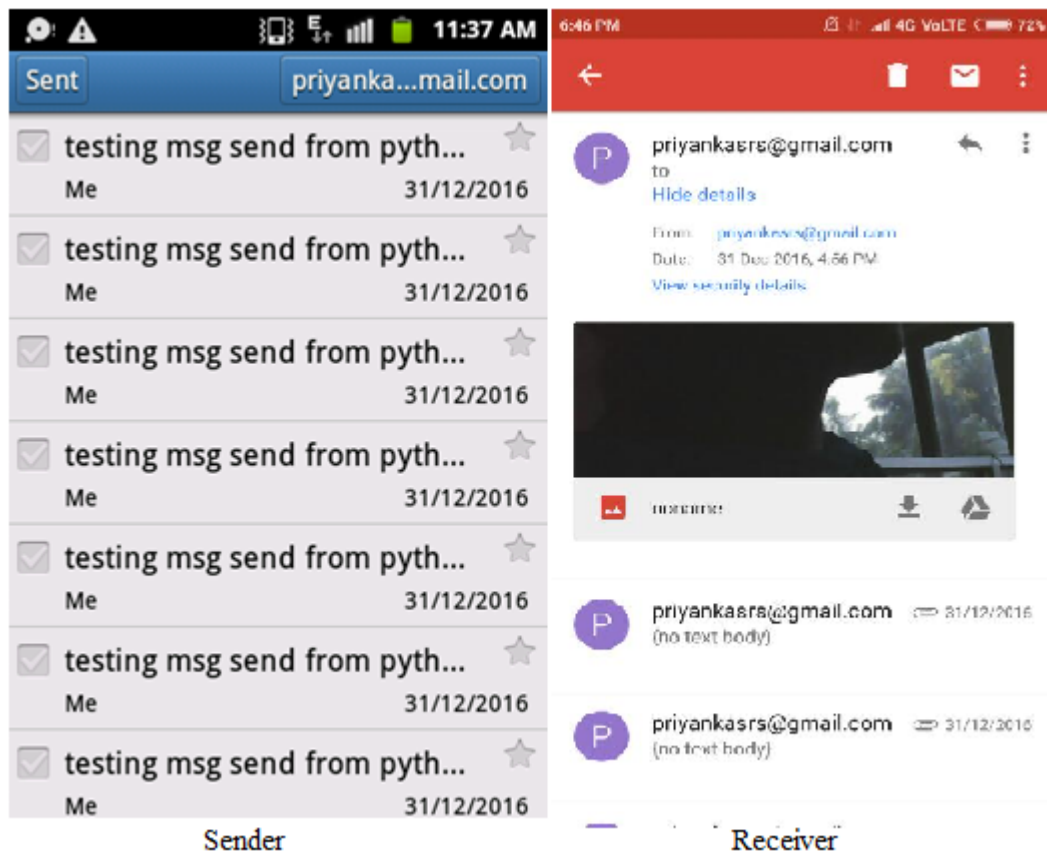


Fig 14: Shows mail sent from priyankaasrs@gmail.com to jyorini30@gmail.com

The mail generation system thus designed can be implemented for government agencies. Such dynamic transfer of information from one remote location to another can be really useful when time is of real need. Along with this the encryption done to protect the image from getting into the hands of anti-social groups is an essential feature for all such highly sensitive data.

9. CONCLUSION

Thus, the main goal of building a Robot monitored by a web-based interface is to provide a completely autonomous robot for the detection of the land mines without any casualties is achieved. This robot can be used in the military applications where risk for human life is the maximum. The given local environment is scanned thoroughly with minimal false alarms and a better accuracy. The prototype is has light-weight, commercially available components which are cost effective which makes it a better solution for mine hunting without involving much manual interaction. The free movement of the robot is also enhanced as well as owing to its weightlessness, the ability to detect mines without detonating them makes this model a suitable choice. The user-interface is minimal and is aimed at satisfying personnel with a non-technical background. The transfer of image using Wi-Fi through electronic mail system achieved by the enhanced play-fair cipher encryption adds the security factor of the robot. The camera module ensures that the images of the environment are captured accurately in case if a landmine is detected. The future scope of the project lies in testing the robot in different environments and scenarios, taking into consideration the weather factors. Also the type of metal detected by the robot could be categorized based on which the demining team could decide upon the severity with which they should apply demining techniques. The overall optimization of the robot in terms of cost effectiveness, accuracy and the algorithmic complexity can also be improved in the future versions of the project.

10. REFERENCES

- [1] Govindaram, B. and Umamakeswari, A. “An autonomous approach for efficient landmine detection and marking using high sensitive robot”, *Int. J. Advanced Intelligence Paradigms*, Vol. 7, Nos. 3/4, pp.280–291, 2015.
- [2] James M. Sabatier , “Advances in Acoustic Landmine Detection”, National Center for Physical Acoustics University of Mississippi1 Coliseum Drive University, MS 38677 USA,2015
- [3] Mark Fisher, Marcus laGrone, Colin Cumming, “Utilization of Chemical Vapor Detection of Explosives as a Means of Rapid Minefield Area Reduction”, Eric Towers Nomadics, Inc.,2002
- [4] Michael O’Connor, Thomas Bell, Gabriel Elkaimand Dr. Bradford Parkinson, “Automatic Steering of Farm Vehicles Using GPS”, 3rd International Conference on Precision Farming, Minneapolis, MN, June, 1996.
- [5] Mohit Kumar, Akshat Aggarwal and Ankit Garg, “A Review on Various Digital Image Encryption Techniques and Security Criteria”, *International Journal of Computer Applications* (0975 – 8887) Volume 96– No.13 , June 2014
- [6] Sudharshan Chakravarthy, S. PrasannaVenkatesan, J. MadhavAnand and J. Jennifer Ranjani, “Enhanced Playfair Cipher for Image Encryption using Integer Wavelet Transform”, *Indian Journal of Science and Technology*, Vol. 9, Issue 39, October 2016.
- [7] Buniyamin N, Wan Ngah W.A.J, Sariff N, Mohamad Z, “A Simple Local Path Planning Algorithm for Autonomous Mobile Robots”, *International Journal Of Systems Applications, Engineering & Development*, Volume 5, Issue 2, 2011
- [8] James Trevelyan, SabbiaTilli, Bradley Parks and Teng Han Chiat, “Farming minefields: economics of remediating land with moderate landmine and UXO Concentrations”, *Journal of Demining Technology Information Forum*, Volume 1, No. 3, 2002
- [9] Atikah Janis, Abdullah Bade, “Path Planning Algorithm in Complex Environment: A Survey”, *Transactions on Science and Technology*, Volume 3, No. 1, 2016
- [10] Pankaj C. Bhope1, Anjali S. Bhalchandra, “Various Landmine Detection Techniques: A Review”, *International Journal of Innovative Research in Science, Engineering and Technology*, Volume 4, Special Issue 6, 2015
- [11] Paddy Blagden, “Mine Detection and the need for new technology, Geneva International Centre for Humanitarian Demining”, *Forum on Physics and Society, American Physical Society Sites*, Volume 31, No. 3, 2002
- [12] Muhammad Zubair, Mohammad Ahmad Choudhry ,Land Mine Detecting Robot Capable of Path Planning, Second World Congress on Software Engineering, IEEE, December 2010
- [13] Nguyen Trung Thanh,HichemSahli, And DinhNhoHao, “Infrared Thermography For Buried Landmine Detection: Inverse Problem Setting”, *IEEE Transactions On Geoscience And Remote Sensing*, Vol. 46, No. 12, December 2008.

11. APPENDIX

PYTHON MAIN CODE FOR GPS MODULE AND BUG ALGORITHM:

```
port = serial.Serial("/dev/ttyUSB0", baudrate=9600, timeout=3.0)
GPIO.setmode(GPIO.BCM)
GPIO.setup(20,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(16,GPIO.OUT)
GPIO.setup(27,GPIO.OUT)
GPIO.setup(6, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(12, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(26, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(19, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(13, GPIO.IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(5, GPIO.IN, pull_up_down = GPIO.PUD_UP)
data=""

while True:
    rcv = port.read(1)
    while(rcv is not 'L'):
        rcv = port.read(1)
    if(rcv=='L'):
        for i in range(0,25):
            rcv = port.read(1)
            data = data+rcv
        variab='L13.539,,192132.086,V,N*7E'+data
        f=open('log.txt','w')
        f.write(variab)
        f.close()

    print data
    if(GPIO.input(13) == 1):
        GPIO.output(27, True)
        print("metal detected at gps value:"+data)
        variab='metal Detected'
        f=open('log.txt','w')
        f.write(variab)
        f.close()
        time.sleep(1)
        print('CAPTURING IMAGE')

        subprocess.Popen( "fswebcam -r 1280x720 /home/pi/chennai/download.png",
                           shell=True )

        time.sleep(1)
        subprocess.Popen("python /home/pi/chennai/mail.py", shell=True)
        time.sleep(1)

    else:
```

```

print ('Nearby no metals present')
time.sleep(1)
variab='Nearby no metals present'
f=open('log.txt','w')
f.write(variab)
f.close()
time.sleep(1)

if(GPIO.input(6) == 1):
print('Obstacle Detected')
variab='obstacle detected'
f=open('log.txt','w')
f.write(variab)
f.close()
time.sleep(1)
print('CAPTURING IMAGE')
GPIO.output(27, True)
subprocess.Popen( "fswebcam -r 1280x720 /home/pi/chennai/download.png",
                    shell=True )

time.sleep(1)
subprocess.Popen("python /home/pi/chennai/mail.py", shell=True)
time.sleep(1)
#right
GPIO.output(22 , False)
GPIO.output(20 , False)
GPIO.output(16 , False)
GPIO.output(17 , True)
time.sleep(5)

# 'left'
GPIO.output(22 , False)
GPIO.output(20 , False)
GPIO.output(16 , True)
GPIO.output(17 , False)
time.sleep(5)

print'forward'
GPIO.output(22 , False)
GPIO.output(20 , False)
GPIO.output(16 , True)
GPIO.output(17 , True)

else:
print('Near By No object Available')
time.sleep(1)
variab='Near By No object Available'
f=open('log.txt','w')
f.write(variab)
f.close()
time.sleep(1)

```

PYTHON MAIL GENERATION CODE:

```
gmail_user = "priyankasrs@gmail.com"
gmail_pwd = "*****"
FROM = " priyankasrs@gmail.com"
TO = [' jyorini30@gmail.com '] #must be a list

GPIO.setmode(GPIO.BCM)

GPIO.setup(25, GPIO.OUT)

GPIO.output(25, False)

time.sleep(1)
msg = MIMEMultipart()
time.sleep(1)
msg['Subject'] = "testing msg send from python"
time.sleep(1)
fp = open("/home/pi/chennai/download.png", 'rb')
time.sleep(1)
img = MIMEImage(fp.read())
time.sleep(1)
fp.close()
time.sleep(1)
msg.attach(img)
time.sleep(1)
try:
    server = smtplib.SMTP("smtp.gmail.com", 587) #or port 465 doesn't seem to
work!
    print "smtp.gmail"
    server.ehlo()
    print "ehlo"
    server.starttls()
    print "starttls"
    server.login(gmail_user, gmail_pwd)
    print "reading mail & password"
    server.sendmail(FROM, TO, msg.as_string())
    print "from"
    server.close()
    print 'successfully sent the mail'
    GPIO.output(25, True)
    time.sleep(2)
    GPIO.output(25, False)
except:
    print "failed to send mail"
    GPIO.output(25, False)
```

PYTHON PACKAGES REQUIRED FOR THE PROJECT:

- a) RPi.GPIO as GPIO
- b) time
- c) serial
- d) subprocess
- e) from subprocess import call
- f) time
- g) os
- h) glob
- i) smtplib
- j) base64
- k) from email.mime.image import MIMEImage
- l) from email.mime.multipart import MIMEMultipart
- m) sys