

**CG1112 Engineering Principles and Practice**

Semester 2 2020/2021

**“Alex to the Rescue”**

**Design Report**

**Team: B02-6A**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Student ID | Sub-Team | Role |
| Zhuang Jianning | A0214561M | Firmware & Software | Algorithm Design & Implementation |
| Kwek Ming Shun | A0206038M | Hardware & Movement | Hardware Design & Movement Calibration |
| Alexander Tan Jun An | A0199267J | User Interface & Wireless Operations | UI design &  Wireless Communications |
| Flores Wraine Melveirich Rivadeneira | A0216014X | Environment Mapping | LiDAR Operations & SLAM |

**Table of Contents**

Section 1: Introduction and System Functionalities …………….…………………………… 3

Section 2: Review of State of the Art ………………………………………………………... 4

Section 3: System Architecture .......…………………………………………………………. 5

Section 4: Component Design ……………………………………………………………….. 7

Section 5: Project Plan ………………………………………………………………………. 9

References…………………………………………………………………………………... 11

**Section 1 Introduction and System Functionalities**

According to the Centre for Research on the Epidemiology of Disasters (CRED), natural disasters have claimed an average of 68,000 lives per year between the years of 1994 and 2013 [1]. In order to reduce this number, faster and more efficient search-and-rescue (SAR) operations are necessary, with the first 72 hours being the “golden period” to locate and rescue survivors [2]. Fortunately, advancements in robotics has led to the increase in use of robots and drones in SAR operations [3]. Not only does this allow for operations in environments hostile to humans, the information gathered from the sensors on the robot can also be gathered and analysed quickly to organise effective rescue operations.

Alex is a robotic vehicle designed as a SAR robot which can be remotely controlled. It will be able to aid in accelerating SAR operations through environment mapping, navigation and victim location.

The essential functionalities Alex would need to achieve its aim are as follows:

1. Remote communications:

Alex is able to receive commands from an operator remotely through the wireless communications capabilities of the Raspberry Pi. The commands are then translated into signals which control the Arduino, Motors and LiDAR.

1. Environment Mapping:

Alex is able to analyse data received from the LiDAR to map out the environment around it in a format recognisable to the operator.

1. Navigation:

Alex is able to move straight, turn left and turn right based on the commands given by the operator. The speed of the movement can also be controlled by the operator. With the mapped environment from the LiDAR, the operator can identify obstacles and avoid them.

1. Identifying Objects

Alex is able to determine the colour of an object in front of it using an LED and Photoresistor. The colour of the object will be relayed back to the operator.

**Section 2 Review of State of the Art**

Alex is a remotely controlled SAR robot whose main objective is to map out a simulated post-disaster environment. The fundamental functions Alex can perform are: (1) Environment Mapping of its immediate surrounding; (2) Movement and Navigation based on the currently mapped environment; (3) Object Identification based on colour. Its key components include a LiDAR, a Raspberry Pi, an Arduino and motors.

The two existing SAR robots chosen to be evaluated are WALK-MAN and Colossus. Understanding their features and components allows for insight to be gained on their strengths and weaknesses. This can then be applied when implementing the hardware and software of Alex to build a more effective SAR robot.

1. **WALK-MAN [4]**

WALK-MAN was invented in 2015 during the DAPPA robotics challenge finals. Roboticists at the Italian Institute of Technology (IIT) had been working to improve the capabilities of the previous version of their custom-made humanoid disaster robot. It has accelerometers, force and torque sensors to balance the body and to allow for human-like movements.

WALK-MAN’s strengths are its ability to perform certain humanoid tasks in a danger prone zone. It is able to use a fire extinguisher, walk through doors and help evaluate its surroundings. However, it also has its disadvantages. The main issues were related to the dimensions and weight of the robot, as well as with the physical performance of the arms. The weight of the robot causes very bulky movement and a limited range of motion. The dimensions of its upper body reduces its capacity to operate within human environments, like passing through doors. The robot is under the command of a human operator for about 80 percent of its actions, with some local autonomous functions related to whole body motion actions and stabilization.

**2. Colossus [5]**

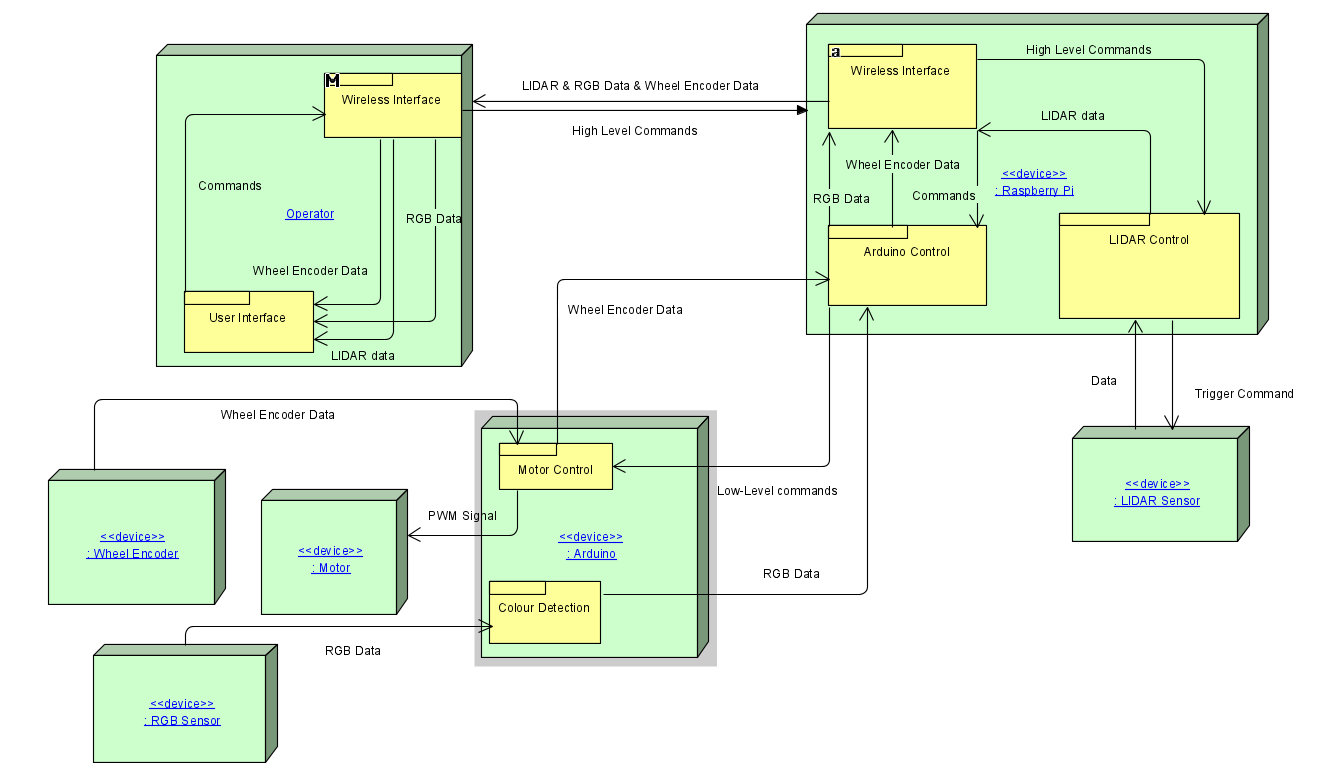
Colossus by Shark Robotics is a remote-controlled robot with a modular design which supports operational responders in their missions. It has a variety of modular options such as water cannons, manipulation arms and smoke extraction fans. This helps in assisting the responders on specific problems like extinguishing fires, removal of obstructions and smoke clearance.

Colossus’ strengths are its versatility from its modular functionalities which range from smoke clearance to extinguishing fires. Moreover, it has high heat resistance which is up to 900°C so it is suited for SAR operations in buildings engulfed in flames. Furthermore, it is low maintenance and can be deployed quickly. However, it has its disadvantages such as its heavy weight of 485 kg, disallowing it to be deployed to fragile, devastated structures as the building may collapse Additionally, it requires high power to operate due to it being a fully electric robot.

**Section 3 System Architecture**

This section will focus on the structural design of the system used to achieve the essential functionalities of Alex. Alex has 3 main control components: the Operator, Raspberry Pi and Arduino which can send commands and receive data from the peripheral devices and sensors. Figure 1 below illustrates the high level components as well as information that passes between them.

The green boxes represent hardware devices while the yellow boxes represent software modules. The connection between the components contains the information that passes between them.



**Figure 1: System Architecture of Alex**

**Operator**

The operator component consists of 2 software modules, the user interface and the wireless interface. They allow the user to send commands remotely to Alex and visualise the mapped environment received. The modules are described in further detail below.

1. **User Interface**

The user interface is where the user interacts with Alex. Both the issuing of commands and the viewing of the data is done here. When a command is issued, it is sent to the Raspberry Pi through the wireless interface where it is converted to a high-level command. For example, a “move forward” command is converted to “rotate right wheel and left wheel forward simultaneously”. When data is received, it is processed and displayed for the operator to view and make the next decision.

1. **Wireless Interface**

The wireless interface sends the high-level commands, received at the user interface over a wireless network to a Raspberry Pi onboard of Alex. Additionally, it receives data such as the LiDAR readings and Wheel Encoder information from Alex.

**Raspberry Pi**

The Raspberry Pi is mounted onto Alex’s chassis and consists of a wireless interface and 2 software modules which control the Arduino and LiDAR.

1. **Wireless Interface**

The wireless interface receives high-level commands from the Operator component and sends it to the appropriate software control module. It also receives LiDAR readings and Wheel Encoder data from the Arduino Control module and LiDAR control module and sends it back to the Operator component.

1. **Arduino Control Module**

The Arduino Control module translates the high-level commands to low-level commands suitable for the Arduino to carry out. This controls the motor control software module on the Arduino. For example, “Move forward” → Move forward 8 encoder ticks at 50% power. Additionally, it receives wheel encoder data from the Arduino and sends it back to the operator’s device.

1. **LiDAR Control Module**

The LiDAR Control module translates high-level commands to low-level commands that can be interpreted by the LiDAR device. It also receives the scanning data from the LiDAR device and sends it back to the operator through the wireless interface.

**Arduino**

The Arduino Uno is mounted onto Alex’s chassis and consists of the Motor Control module and the Colour Detection module.

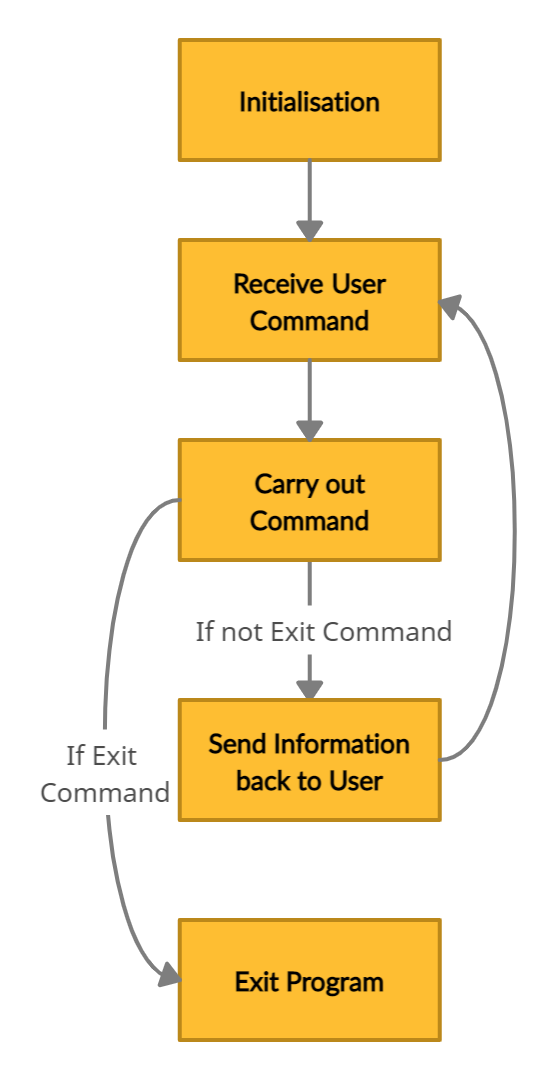
1. **Motor Control**

The Motor Control module outputs a PWM signal at particular pins to rotate the motors according to the movement command received. Hall effect sensors are attached to the motors with a magnet to detect the rotation of the wheels. For each unit of rotation, pulses of current are sent to the Arduino Uno. This information is used to determine how far Alex has travelled and how much Alex turns when move commands are given. The data is also used to calibrate the movement commands and can be sent back to the operator to be displayed.

1. **Colour Detection**

The Colour Detection module outputs a PWM signal to the 3 RGB anodes to obtain different colours. The light is then reflected by the object we wish to detect onto the photoresistor. The resistance of the photoresistor is detected and sent back to the Colour Detection module which processes the data and determines the colour of the object. The colour detected is then sent back to the operator.

**Section 4 Component Design**

This section will focus on the overall algorithm used to remotely control Alex. It will also explain the flow of control between the User, Raspberry Pi, Arduino and Lidar. The high level steps for the algorithm used by Alex will be described first, followed by a further breakdown and elaboration on each step. Figure 2 below illustrates the overall algorithm.

High Level Steps:

1. Initialisation
2. Receive User Command
3. Carry out Command
   1. If Command is Exit, skip to Step 5.
   2. Else, continue to Step 4.
4. Send Information back to User
   1. Return to Step 2.
5. Exit Program

Further breakdown:

Step 1. Initialisation

1. Power on all devices.
2. Operator starts the program, which initialises the wireless interface.
3. Remote device and Raspberry Pi’s wireless interfaces perform a handshake.

**Figure 2: Overall Algorithm**

1. Upon receiving the handshake, the Raspberry Pi initialises the Arduino and LiDAR control modules.
2. The communication protocol between Raspberry Pi and Arduino uses a baud rate of 9600 and an 8N1 frame format.

Step 2. Receive User Command

1. Commands are entered through the user interface and parsed to determine the corresponding high-level commands.
2. The high-level command is sent through the wireless interface between the remote device and Raspberry Pi and returns an acknowledgement response.
3. The corresponding control modules for the Arduino or LiDAR are invoked.

Step 3. Carry out Command

1. The Arduino and/or LiDAR control modules receive the high-level commands and return an acknowledgement.
2. The high-level commands are translated to low-level commands for the peripheral devices.
3. If the command is an exit command, the program ends gracefully.
4. Else, data collected by peripheral devices are sent back to the remote device.

Step 4. Send data back to User

1. If there is data collected by the peripheral devices, they are sent back to the control modules.
2. The control modules send the data back to remote device through the wireless interface.
3. The data is formatted and presented in the user interface.
4. Control returns to the user interface and the Operator decides the next action to take.

Step 5. Exiting the Program

1. If the command received was an exit command, the program ends gracefully.

**Section 5 Project Plan**

Recess week:

* Understanding the project requirements
* Doing a brief overview of various sections

Week 7:

* Understanding basic lidar functions
* Understanding communication protocols

Week 8:

* Understanding how to control Alex remotely
* Understanding secure communication to and fro Alex

Week 9:

* Understanding more about SLAM using Lidar
* Understanding about power management in the overall system
* Starting on the final report

Week 10:

* Connecting the different modules with one another to create the desired architecture
* Organizing the library code for each individual module to prepare for integration
* Adding and preparing for final report submission

Week 11: (Final Report Draft)

* Finalise Firmware, software and hardware design and prototype of Alex should be established.
* Basic Commands of movement and Lidar Mapping features should be up
* Working on debugging any integrated firmware code
* Planning for Additional Functionality
* Finalising final report

Week 12:

* Basic requirement of project should be finalised
* Implementing additional functionalities
* Debugging and finalising final implementation

Week 13: (Final Evaluation Day)

* All systems should be up and running
* Testing leading to minimize bugs
* Alex robot is ready for Grading

Detailed Timeline

Software

1. Overall Algorithm

* Initialisation : All hardware should be online.
* Receive User Command - User command is parsed through the User Interface and sent to the operator to execute the commands.
* Carry out Command - The Arduino/Pi control modules receive the commands to be executed and push it to the hardware for execution.
* Send data back to the user - Any incoming data i.e. LIDAR, RGB sensor, motor, will be sent back to the operator through the Pi wireless interface.

1. Pi Algorithm

* Initialising - Booting up the Pi and securing it to the operator.
* Receiving command from user - User command is parsed through the User Interface and passes from the operator to the Pi through the wireless interface
* Sending information back to the user - Using UART to communicate with the Arduino to which will handle the appropriate response to implement.

1. Operator Algorithm

* Initialising - Booting up the program, ensuring link between Pi and User Interface.
* Receiving command from user - Constantly waiting for response from User interface. Once commands are read, the operator sends the appropriate data to the Pi for execution.
* Receiving packets from the Pi - Listen for data sent back from the Pi, which will trigger the appropriate response/

1. User Interface features - To prepare a clean display for interacting with the program.

Hardware

1. Arduino Algorithm

* Initialisation - The arduino is booting up correctly.
* Receiving Commands - Preparing the Arduino to check for new commands communicated by the Pi, read it and have the appropriate commands carried out
* Carrying out Commands - On receiving command, it writes the appropriate values to the various output registers to control the motor while receiving data from the movement of the wheel.

1. Communication Protocol - Setting up the Uart communication between Arduino and the Pi as learnt from the studio.
2. Hardware Design - Configuring the final design and components needed for the robot to be assembled and finalised.

**References**

[1] “The human cost of natural disasters: a global perspective | PreventionWeb.net.” <https://www.preventionweb.net/publications/view/42895> (accessed Mar. 03, 2021).

[2] “Why the First 72 Hours After a Disaster Are Critical.” <https://www.primalsurvivor.net/why-the-first-72-hours-after-a-disaster-are-critical/> (accessed Mar. 03, 2021).

[3] “Search and Rescue Robots Market | Growth, Trends, and Forecasts (2020 - 2025).” <https://www.mordorintelligence.com/industry-reports/search-and-rescue-robots-market> (accessed Mar. 03, 2021).

[4] “The humanoid robot WALK-MAN for supporting emergency response teams | EurekAlert! Science News.” <https://www.eurekalert.org/pub_releases/2018-02/iidt-hrw022118.php> (accessed Mar. 03, 2021).

[5] “Colossus - ROBOTS: Your Guide to the World of Robotics.” <https://robots.ieee.org/robots/colossus/> (accessed Mar. 03, 2021).