CG2111A Engineering Principle and Practice

Semester 2 2021/2022

"Alex to the Rescue"

Design Report

Team: B01-2A

Name	Student #	Sub-Team	Role
Tan Wei Li	A0216235M	1	Hardware
Bui Phuong Nam	A0244121X	2	Software
Anderson Leong Ke Sheng	A0223111E	2	Software
Chan Ee Hong	A0233898L	2	Sublead
Aung Phone Naing	A0234363J	1	Lead

Section 1 System Functionalities

A robotic vehicle, hereby referred to as Alex should be built for search and rescue operation purposes. The motors, embedded controllers, power bank, chassis, and Lidar provided should be used without any modifications.

The embedded controllers provided are a Raspberry Pi, connected to an Arduino Uno. Tele-operation is carried out by having the laptop communicate with the raspberry pi to command Alex.

Hence, Alex should be teleoperated from a distance away and map a maze about $3m^2$ in the area as it moves. It should also be able to detect people in the map that is produced by the Lidar mounting and be able to navigate the maze from a designated start point to a designated endpoint. The minimal requirement for Alex is that it should be able to go straight, or turn left/right. There may be unfriendly terrains such as a hump in the maze and Alex should be able to navigate over it and any forms of collision should be avoided.

Section 2 Review of State of the Art

Snakebot

Developed by Carnegie Mellon University, this modular snake-like robot can move on land and underwater to explore places that are difficult for humans to access. It has a camera at the front of the robot that streams live video back to the teleoperator. It moves across land by twisting and turning different joints in its body, and can autonomously adjust its movement to climb vertical obstacles based on the pole's diameter. To move underwater, it has small propulsion fans on each of its modules, allowing precise orientation control in confined spaces. It is lightweight and can reach high speeds underwater. It is easy to learn how to operate the snakebot, as it has many LED indicator lights, a robust control box where live video is streamed back from the robot, and a lightweight compact tether.

Strengths of snakebot include being able to fit in tight spaces (under debris, the hull of ships, nuclear reactors) to relay critical information which can save lives, time, and money, lightweight (easy to transport and deploy), and being able to traverse both on land and underwater. Weaknesses include being unable to carry heavy payloads, unable to perform complex tasks (like lifting or moving objects), and being unable to move as fast as other robots over open terrain.

Colossus

This fire-fighting robot by Shark Robotics helped the Paris Firefighter Brigade put out the fire at Notre Dame cathedral. It weighs about 500kg, and has treads as wheels to manoeuvre over uneven terrains such as stairs and debris. A water hose can be connected to the back of Colossus to allow it to spray more than 2000 liters of water every minute. It is also strong enough to transport heavy fire fighting equipment and carry victims out of dangerous areas. Colossus has a 360-degree high-definition thermal camera that allows teleoperators to have a good idea of what is happening.

The strengths of Colossus include the ability to carry heavy payloads, the ability to withstand extreme heat, and the ability to navigate across urban obstacles like stairs and buildings. Weaknesses include being hard to transport due to its heavyweight, and not being as agile and fast as smaller robots.

Section 3 System Architecture

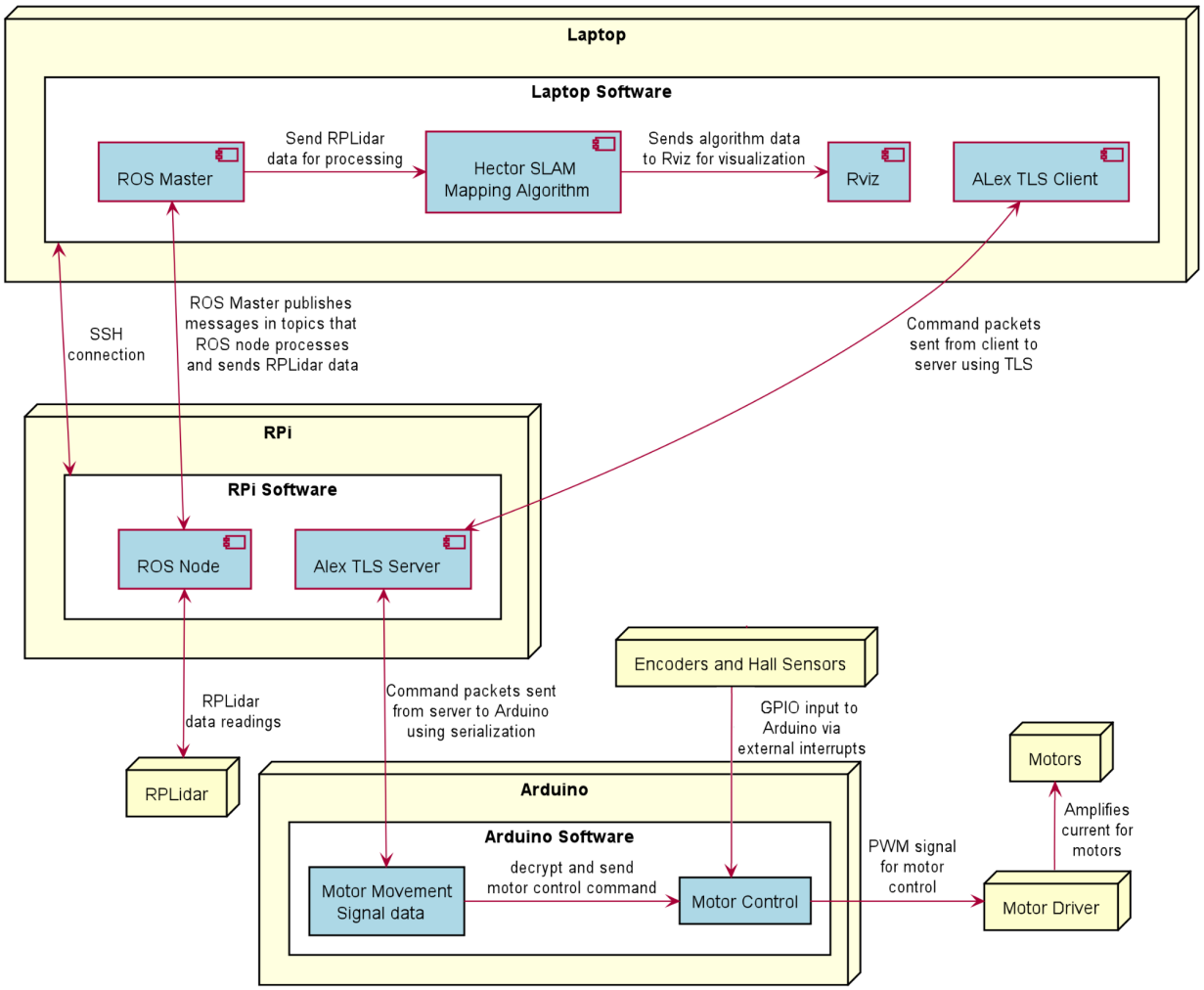

Figure 1: System Architecture of Alex

The above figure depicts the high level components of Alex. Yellow blocks represent hardware, in which there may be relevant software components to be emphasised which are indicated in blue rectangles.

Section 4 Component Design

High-Level Steps to solve the problem:

1. Initialization
2. Receive user command
3. Carry out command
4. Mapping
5. Repeat step 2 until navigation is over.

Further breakdown:

Step 1: Initialization

a. RPi sends a predetermined packet to Arduino.
b. Arduino reads the received packet and checks for errors such as BadPacket or BadChecksum.
c. Arduino replies to RPi if the received packet is good or erroneous.
d. In the case of a good packet, Arduino starts listening for another packet from the RPi, otherwise, the system should be rebooted.

After the successful handshake between RPi and Arduino, the Lidar will be started and sent its initial scan to Pi.

Step 2: Receive User command

a. Parity byte checksum and magic number checksum to verify the integrity of transmitted packet
b. If both checks passed, the device proceeds to check if the sent command is valid
c. If all three checks are passed, the command will be executed. Otherwise, an error message will be returned.

Step 3: Carry out command

a. For Alex's movement, the Arduino will check its list of commands to verify if the command sent is valid.
b. In the case of a valid command, the Arduino will execute the command and return the "g" command to RPi.

Step 4: Mapping

a. Lidar data will be continuously transmitted back to the ROS Master on the laptop
b. ROS Master sends this data to the Hector SLAM Mapping algorithm
c. Results of SLAM are sent to RViz to be visualised by teleoperator

Section 5 Project Plan

Week 8 (7th Mar - 13th Mar):
Both subteams will work together to assemble Alex. We will first read through the given Alex assembly guide thoroughly to avoid mistakes. We will discuss the layout of our electronics and hardware. We will wire the electronics as per the guide given. Following the studio activities, we will set up the Arduino libraries on the RPi, set up and test the wheel encoders, and set up Alex's motor control routines together.

Week 9 (14th Mar - 20th Mar):
Following the studio activity for this week, both subteams will work together to perform the initial movement calibration of Alex's movement, which is dependent on our wheel encoders. This is done by adding code to the existing Arduino file as per the studio activity instructions. We will also obtain movement data from Alex. Next, we will try out the serial communication protocol given to us for serial communication between the RPi and the Arduino.

Week 10 (21th Mar - 27th Mar):
Following the studio activity for this week, both subteams will work together to build the Alex TLS server on the RPi and the Alex TLS client on the laptop. We will also learn more about network debugging tools like Netcat, which subteam 1 may need for implementing ROS connection soon.

Week 11 (28th Mar - 3rd Apr):
Subteam 1 will also conduct preliminary tests using code from previous weeks to ensure that hardware components are still correctly connected. Subteam 1 will also need to figure out how to implement ROS on the laptop and connect with the ROS node on the RPi in order to process RPLidar data for RViz on the laptop, and test this connection. Based on preliminary knowledge, we presume that one viable method is to have a virtual image of Ubuntu. Hence, subteam 1 will need to research how to set up the Ubuntu image for connection with the RPi via ROS also.

Subteam 2 will focus on software improvement on the existing codebase. They will implement the code required for using ultrasonic sensors with Alex, bare-metal programming where possible, and optimising the code for Alex's movement via the command-line interface. These changes will be done using Git version control. They will need to test the new code, which can still be done with existing code from the previous week or after the ROS connection between the laptop and RPi is established.

Both subteams will have the shared responsibility of calibrating Alex's control and movement, seeing where improvements can be made to optimise this. Also, both subteams will start preliminary discussion together to assign tasks for the design report.

Week 12 (4 Apr - 10 Apr):
Subteam 1 will debug the existing improvements from subteam 2 by integrating what is available into Alex first. This also tests the ROS connection to ensure that it is working. Subteam 1 will also work on improving cable management, optimising placement of components of Alex and preparing for hardware integration of ultrasonic sensors.

Subteam 2 will continue their effort in bare metal programming and optimising code for alex movement.

Members from both subteams will start on their assigned tasks for the design report. Both subteams will also attend the demo run to review Alex's current performance and rectify any major issues as soon as possible.

References

https://builtin.com/robotics/rescue-robots

https://www.cmu.edu/news/stories/archives/2021/april/snake-robot.html

https://www.washingtonpost.com/technology/2019/04/17/firefighters-had-secret-weapon-when-notre-dame-caught-fire-robot-named-colossus/