**Programming Things**

**Search and Rescue [Assignment 1]**

**Joel Halford**

**Task Breakdown**

**Task 1:**

Task one was completed without any issues. The GUI used was a library for Processing called G4P which has a builder which allows a quick and easy creation of a GUI.

**Task 2:**

Task 2 and 3 caused a bit of trouble. Initially I was only using one threshold for all the sensors but soon found out that a better way of getting around this would be to have the middle 4 sensors set to a much lower threshold, which gives the middle sensors time to detect the black border. For this task I used the ZumoReflectanceSensorArray and the QTRSensors which are built-in libraries.

**Task 3:**

Task 3 was a handful. I wanted to set up the Gyro/Compass as it seemed it would help for future tasks but could not get it to work. After multiple days of failed attempts, I scrapped this idea and coded the turns to as close to 90 degrees as possible using a specific turn speed and turn duration.

**Task 4:**

For task four, for example, when sending "rRo" for right room, I was bumping into issues since I was reading characters and now I had to also read strings. I switched everything into strings, which worked but there was a big delay when pressing buttons. Instead, I sent strings as individual characters and nested them inside IFs. While a little messy, it seemed to do the trick. Par this issue, everything else ran smoothly.

**Task 4A and 4B:**

For task 4A, I created an object array of corridors, holding a CorridorID, RoomID and objects. This didn't work as I may need more than 1 room per corridor, so I made a second object array for rooms. The part I struggled with the most was managing to keep all the data easily accessible, i.e. knowing if I have been inside a particular corridor. To solve these programs I made a number of arrays. the data[] array keeps track of left and right movements, left and right corridors and left and right rooms. This way I can check if the last two elements in the array are equal (i.e. leftTurn, leftTurn) to know if I am going back into a corridor.

**Task 4C:**

I believe, since I set up my code in a particular way, covering (close to) each step of its journey, task 4C wasn't too difficult. Luckily, G4P allows to set buttons to enabled/disabled. To achieve this, I first check if the last two elements in the data array are equal, confirming that the corridor is infact one we have been down, read the first character of the CorridorID string (L/R) and disable the opposite button.

**Task 5:**

Since I had prior trouble with Gyro/Compass and was delayed with other issues, I have not attempted this. I believe doing task 5 would require me to change a lot of my current code, due to turning not being very accurate. The robot, at times, needs some assistance and task 5 does not allow for this.

If I had time, I would get the compass working and use this to perform perfect angles and also track data more easily.

**Acknowledgements**

**Ultrasonic sensor**

Source: Trollmaker.com

This allowed for the setup for the ultrasonic sensor. I just needed to change the distance and what happens when an object is detected.

**Processing.org**

Processing.org has been a great help. I struggled a little at first and sometimes had the odd question and processing.org was always there to help. An example of this would be what you can do with serials.

**G4P (GUI for Processing)**

Source: Peter Lager

Peter Lager provides an easy to use GUI builder which helped tremendously. All I had to do was add and customise features to the GUI inside of a visual builder.

**Folder Contents**

**/libraries**

Contains some libraries being used for this project.

**/main**

Main contains my main .ino file. This is the file that must be uploaded to the Arduino.

**/media**

Contains a video of the zumo on a test track.

**/RobotGUI**

Contains the GUI and where the data stored.

**Hardware Used**

1x Zumo Robot

1x Arduino Stackable Headers

1x Arduino board

1x xBee shield

2 x xBees

1x xBee USB adapter

**Software Used**

**XTCU**

Allows use with xBees

**Arduino**

Allowed me to code the zumo and upload to it.

**Processing 3**

Allowed me to create the GUI, interact with the zumo wirelessly and store data regarding the zumo’s readings.