**Programming Things Zumo Report:**

# **Task 1:**

## How I have achieved this task:

Task 1 was the simplest task for this assignment, the manual movement control of the Zumo was achieved by having 5 different motor speed setters and putting them inside of a switch statement, this takes the 5 different button inputs from the GUI and uses them to determine which direction to move the Zumo.

## How to use this task:

The ↑ ↓ buttons send the robot forward and back respectively. The →← buttons rotate the robot clockwise and anti-clockwise, respectively. The stop button will stop the wheel motors. Utilising all these buttons together gives you full control of the robot allowing you to move it freely.

Box and whisker chart

Description automatically generated

## Resources used:

* I used the “LineFollower” Arduino example to learn how to set motor speed.
* I used the “form1.cs” code from session 5-6 on blackboard to learn how to create and open ports for the GUI code.

# **Task 2:**

## How I have achieved this task:

Task 2 is also relatively simple. I achieved this task by implementing a boolean that works as a flag for Zumo autonomous control. It is set to true by pressing the “GO!” button and set to false by the Zumo hitting the end of the corridor.

## How to use this task:

By pressing the GO! Button the Zumo robot will begin its auto navigation staying inside the walls and stopping when it reaches the end of the corridor.

A picture containing text

Description automatically generated

## Resources used:

* I used the “LineFollower” Arduino example for the code for following the corridor and avoiding the walls.
* I used the “BorderDetect” Arduino example to learn how to use the Zumo robots built in line detector.

# **Task 3:**

## How I have achieved this task:

Task 3 is a combination of Task 2’s automatic navigation and task 1 turning code. To make sure I was meeting the requirements of the specification and to make sure that the corner turns were all the correct rotation angle I used the Zumo robot’s onboard gyroscope.

## How to use this task:

The “GO!” button will begin the Zumo robots auto navigation, once the Zumo reaches the end of the corridor (The textbox in the GUI will inform you when this is true) you may press the “L” and “R” buttons to rotate the Zumo robot in that direction. Then use the “C” button to continue your path. Repeat these steps until you have reached your destination.

Graphical user interface

Description automatically generated

## Resources used:

* I used the <https://www.c-sharpcorner.com/uploadfile/eclipsed4utoo/communicating-with-serial-port-in-C-Sharp/> website to learn how to receive the Zumo sent messages and utilise them in the GUI text box
* I used the “MazeSolver” Arduino example and to learn the code for using the onboard gyroscope to turn specific angles. (I had to add TurnSensor.cpp and TurnSensor.h)

## Task 4:

Task 4 is just an extension of task 3. By removing the continue button and just having the Zumo automatically continue auto navigation after a turn is made, I was able to complete this task.

# **Task 5:**

## How I have achieved this task:

For this task I require the Zumo to differentiate between regular auto navigation through corridors and navigation inside of a room to scan. To achieve this, I created two separate flags that switch on an off depending on whether the Zumo is being instructed. To make sure I am meeting the specification I thought it would be best if the Zumo reversed back out of the room automatically after it is done inside the room, with no light sensors on the rear of the robot I used the wheel encoders to achieve this. While entering the room the robot will begin measuring the wheel encoders, once the Zumo is finished in the room it will reverse the same amount of wheel encoders to return to its current position. I used a 2d array to store the room number and properties for the Zumo to use in later tasks.

## How to use this task:

The new buttons on this GUI are the “STOP!” and “ENTER ROOM MODE!”, to search a room you must use the “STOP!” button outside of a room, then use the “ENTER ROOM MODE!” button to signal to the Zumo that it is outside of a room. From this point you can use the “L” and “R” buttons to turn into the room.

Graphical user interface

Description automatically generated

## Resources used:

* I used the “LineAndProximitySensors” Arduino example to learn how to use the Zumo robot’s onboard proximity sensors.

# **Task 6:**

## How I have achieved this task:

Task 6 involves ignoring the users commands while the Zumo is backtracking back to the T-junction, I accomplished this in my code by using encoders again to measure the distance from the T-Junction to the end of the corridor. The robot does a 180 turn to turn around, then to make sure the Zumo ignores commands I temporarily close the port communication, once the encoders indicate we are past the T-Junction I then re-open the port to allow commands again.

## How to use this task:

To backtrack once you reach the end of a corridor after turning past the T-Junction press the “BACK!” button. The textbox will inform you that the controls are locked, any commands sent before the Zumo reaches the T-Junction will not be registered. The textbox will inform you once the Zumo passes the T-Junction and controls will be returned.

Graphical user interface

Description automatically generated

# **Task 7:**

## How I have achieved this task:

For the returning home code, I knew that I would need to record every individual “Step” the robot makes. I used a 2d array for this, each item in this array was another array that contained the encoder distance for that step, whether a room was checked on this step (it stores the room number only if the room was populated) and whether a turn was made on this step. Iterating through this array backwards allowed me to retrace the Zumo robots’ steps and recheck the rooms that were populated when checked the first time. I created a check for when the Zumo rescans a room that was previously occupied, when this is done, I switch on the LEDS and the Zumo buzzer should beep continually until it reaches the end of the maze.

## How to use this task:

To start the Zumo robots return home, just press the “HOME!” button once you have reached the end of the maze (or before the end if you would like to return early )

Graphical user interface

Description automatically generated

## Resources used:

* I used the “BuzzerBasics” Arduino example to learn how to make the Zumo play sound.
* I used the “BlinkLeds” Arduino example to learn how to use the Zumo LEDS.