**What I have achieved**

**TASK 1-6 / BASIC VERSION CONTROL / BASIC GUI**

**Task 1: Manual control of the Zumo**

My manual control of the zumo focused around simply taking the option value sent from the GUI and passing it through a switch statement which then ran the appropriate function for example “w” would increase the speed of both motors, thus moving the zumo forwards.

My manual control featured left and right turning:

A picture containing text

Description automatically generated

As well as a search room, horn function and a force stop button, as you will see in my demo video:

Text

Description automatically generated with low confidence

**Task 2: Autonomous control of the Zumo**

Graphical user interface, text, application

Description automatically generatedAuto control starts off from the very start of the program, as the user must activate calibration mode, which calibrates the line sensors used for auto control:

You can see in program that it moves from side to side whilst calibrating, getting a good reading of the surrounding area, this needs to be done on an open plain not including the black lines.

After calibrating I take the highest values gotten from the readings and save them in my calibrateData array to be able to compare it too later.

Graphical user interface, text, application

Description automatically generated

Here is the code that actually keeps it within the blacklines, sensorvlaues being the current reading found by the sensors and the calibrated data being whaat we saved arlier to compare it to. Since the black lines are solid black we can add 100 to the data we gathered earlier to cancle out any discrepancies or anomalies found on the papar. When sensors hit these vlaues the alternate motor to the sensor activates untill it has returned to being white.

**Task 3 and 4: Turning Corners**

To check if the zumo is at a corner I simply check if either both the sensors are above the threshold, or if simply the middle sensor has been triggered.

When this happens, it changes the mode variable 2 telling the program that we are now at a corner mode, and we need to wait for the user’s action.

Graphical user interface, text, application

Description automatically generated

Within our Conroe control function, we can single whether the corner is a left or right and the zumo will position accordingly, I am doing this using the motor encoder library which allows for a more accurate movement abilities, as the zumo motors can vary in power depending on man things, such as battery life. The motor encoders also allow the motors to be more in sync as sometimes the two motors have different power levels all together.

**Task 5: The Zumo searches a room**

I press p in the GUI to signal that there is a room to be searched by the zumo, this activates search mode. The zumo then waits for the user to press either L or R to indicate which side the room is on

Graphical user interface, application

Description automatically generated

It then goes into the room and scans it, running a function whilst the zumo is looking left and right, if that function ever returns true, we then play a buzzer and indicate to the GUI that a person has been found.

**Task 6: The T-junction**

The T junction works the same as a corner so you can turn left or right, then at the end of the corridors you will be able to press z which will allow you to be a 180 and continue with the other side of the t junction. Then when it reaches the other one it will end the program and tell you how many people have been found.

why and how you resolved (or attempted to resolve) key issues

My main issues resolved around turning the zumo, at first, I tried using the gyroscope to determine a turn. I discovered quickly that the gyro sensor in the zumo is terrible, It worked fine when only using the gyro and getting a reading, but once implementing it into the code It would always move over the amount I wanted.

After some research I determined it was due to the gyro not working in conjunction with other operations from the zumo(maybe the motors). After that I decided to disregard the gyro sensor all together and use the motor encoders.

To make sure the line sensors worked well I decided to use a calibration technique, this allows for some leeway in terms of the colour of the matt, as some parts could be darker, but I don’t want it to stop randomly thinking it’s at a wall.

GUI development

My GUI development was done within processing using JAVA, processing has good integration with the serial port from the zumo and giving the zumo Inputs gave no issues. I had a problem at first using python which led to incognisant moving probably caused by the sending of the data through the port, as python can be bad at that sort of thing.

My GUI only really featured a few labels, which changed based on what the zumo was doing, although giving clear instructions to how to control the zumo when user input is needed.

**CONTROLS MANUAL MODE**

**Step 1**

Text

Description automatically generated

Press C to start Calibration and wait for completion.

**STEP 2**

Text

Description automatically generated

It will start in auto mode, if you are not ready for this press k for manual controls.

**STEP 3**

Text

Description automatically generated

MANUAL MODE CONTROLS:

W ---- MOVE FORWARDS

A ---- MOVE LEFT

D ---- MOVE RIGHT

S ---- MOVE BACKWARDS

Q ---- TURN LEFT 90D

E ---- TURN RIGHT 90D

X ---- EMERGANCY STOP

Z ---- TURN AROUND 180D

L ---- SEARCH A ROOM LEFT

R ---- SEARCH A ROOM RIGHT

V ---- PLAY HORN

O ---- ACTIVATE AUTO MODE

**CONTROLS AUTO MODE**

**STEP 1:**

Make sure you have calibrated on an appropriate surface and then drive in manual to the start.

Text

Description automatically generated

Press O when at the start to start the auto mode.

**STEP 2:**

You will drive until you hit the first wall, you will be greeted by the following.

Text

Description automatically generated

CONTROLS WALL MODE

L ---- TURN LEFT 90D

R ---- TURN RIGHT 90D

Z ---- TURN 180(If at end of corridor)

C ---- Continue search in Auto

STEP 3:

To indicate you are at a room press P to stop

This will activate search mode

Text

Description automatically generated

L ---- SEARCH ROOM LEFT

R ---- SEARCH ROOM RIGHT

C ---- CONTINUE WITH AUTO

**STEP 4:**

Completed Mission

A picture containing text

Description automatically generated