Future Car Series:
The Design of a Mapping and Exploration Mobile Robot.

Final Report

K.L. Jones
12104613

Submitted as partial fulfilment of the requirements of Project EPR400
in the Department of Electrical, Electronic and Computer Engineering
University of Pretoria

November 2016

Study leader: Mr Lijun Zhang

Part 1. Preamble

This report describes the work I did in designing a mobile robot that explores and maps a complex maze independently.

Project proposal and technical documentation
This main report contains a copy of the approved Project Proposal (Part 3of the report) and technical documentation (Part 5 of the report). The latter provides details of the hardware diagrams, user guide, schematics, and software code. I have included this appendix on a CD or USB that accompanies this report.

Project history
The Future car series project was started in 2016 and therefore there is no previous project for this project to follow on from.

Language editing
This document has been language edited by a knowledgeable person. By submitting this document in its present form, I declare that this is the written material that I wish to be examined on.

My language editor was Mr. Greg Lavagna .

Please refer to the end — 07-11-2016
Language editor signature — *Date*

Declaration

I, Kirsty Leigh Jones understand what plagiarism is and have carefully studied the plagiarism policy of the University. I hereby declare that all the work described in this report is my own, except where explicitly indicated otherwise. Although I may have discussed the design and investigation with my study leader, fellow students or consulted various books, articles or the internet, the design/investigative work is my own. I have mastered the design and I have made all the required calculations in my lab book (and/or they are reflected in this report) to authenticate this. I am not presenting a complete solution of someone else.

Wherever I have used information from other sources, I have given credit by proper and complete referencing of the source material so that it can be clearly discerned what is my own work and what was quoted from other sources. I acknowledge that failure to comply with the instructions regarding referencing will regarded as plagiarism. If there is any doubt about the authenticity of my work, I am willing to attend an oral ancillary examination/evaluation about the work.

I certify that the Project Proposal appearing as the Introduction section of the report is a verbatim copy of the approved Project Proposal.

_____ — 08-11-2016
K.L Jones — Date

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

PC	Personal Computer
DSP	Digital Signal Processor
PIC	Programmable Intelligent Computer
DSPIC	Digital Signal Programmable Intelligent Computer
IC	Integrated Circuit
UART	Universal Asynchronous Receiver Transmitter
PWM	Pulse Width Modulation
ASCII	American Standard Code for Information Interchange
LED	Light Emitting Diode
USB	Universal Serial Bus

Part 2. Summary

.

Future Car Series: The Design of a Mapping and Exploration Mobile Robot

K.L Jones and Mr Lijun Zhang
Department of Electrical, Electronic and Computer Engineering
University of Pretoria

***Abstract*—this report describes the work done on the design and implementation of a mobile robot that drives through a maze, maps the maze in the form of a PC (Personal Computer) 2D (Two Dimensional) display, and calculates the shortest route from an entrance to an exit in the maze.**

What has been done?

- **A compact mobile robot has been designed and built to navigate the maze.**
- **An algorithm to map the walls of the maze to a PC display has been completed.**
- **An algorithm that allows the mobile robot to navigate the whole maze has been completed.**
- **An algorithm to find the shortest route from the entrance to exit has been completed.**
- **A remote control has been designed and built to control the movement of the mobile robot wirelessly.**

What has been achieved?

- **The size of the maze is ten times larger than the mobile robot.**
- **The mobile robot moves around walls with a minimum distance of 2cm**
- **The mobile robot responds correctly to instructions from the remote within 1 second.**

I. PROBLEM IDENTIFICATION

The future car series project has been split into smaller projects such as speed and steering control of a self-driving car, automatic parking control of a self-driving car, and mapping and exploration with a mobile robot. This project's focus is on the mapping and exploration of a mobile robot through a maze. The link between this project and the future car series is found in the need to create a mobile robot that runs automatically or manually if desired. The mobile robots' algorithms will need to be modified for a vehicle that drives through road situations and not a maze, as a separate project

The prerequisite of this project is to build a mobile robot that will run through a complex maze automatically. The mobile robot must save information of the complex positions of the walls, process the information, and send the information to a PC to display the maze in 2D format. The size of the maze must be at least ten times larger than the mobile robot and the mobile robot must calculate the shortest route from one of the entrances of the maze to a predetermined exit. This shortest route is displayed in the 2D representation of the maze and is demonstrated by the mobile robot through the maze. The mobile robot must be able to be controlled with a remote control, if desired, and stop before driving into a wall.

Accuracy is a large factor in this project. Both the display and shortest route outputs depend on the accuracy of the mobile robots movements, the accuracy of the sensor system, and therefore the correct decisions made from these distance readings. The accuracy of the remote control depends on the wireless communication system and the correct interpretation of the instruction sent from the remote. By design the maze must be ten times larger than the mobile robot. To satisfy battery life and travel time requirements, the maze is desired to be small, thus, the mobile robot is designed to be compact. The Methods' section describes how these problems were overcome to create a system that worked accurately for the desired results.

II. METHODS

The approach to the project was to split the project up into five parts;

- the maze,
- the mobile robot,
- the display,
- the remote control and,
- the shortest route algorithm.

The functional diagram in Figure 1 shows how the five parts are related in the project.

The movement of the mobile robot was designed first, FU1.4. Servo motors were considered, but after testing the motors they were found to create errors that were

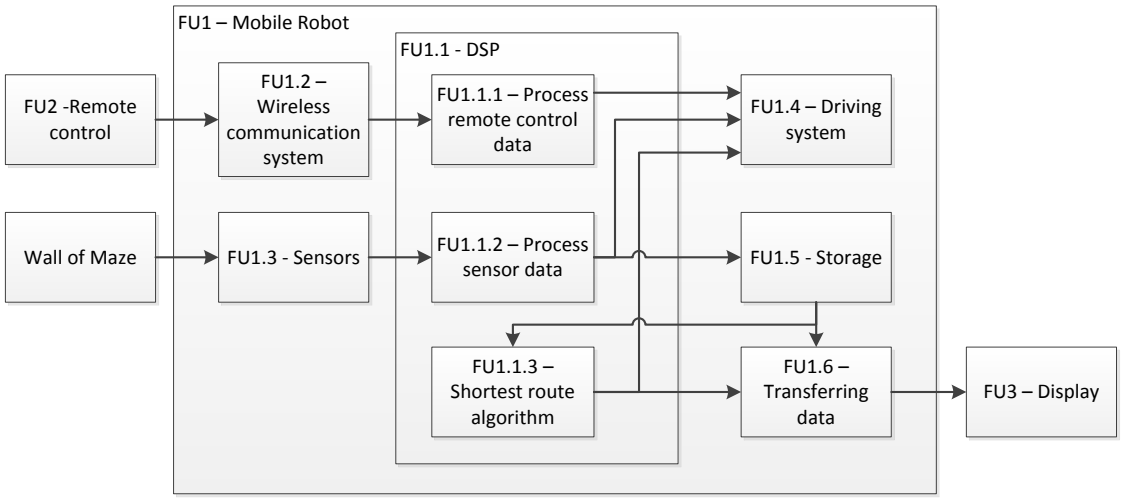


Figure 1: Functional diagram of the project

undesirable. So stepper motors were chosen as a more viable option for the movement system. The stepper motors have a step angle of 5.125° and therefore allowed for increased accurate control over the movements when half-stepping was implemented to decrease the step angle to 2.562° [1]. The turning circle of the mobile robot must be kept to a minimum in order to monitor the movements and position of the mobile robot in the maze. The stepper motors and wheels connected to the motors were thus set up to be opposite each other, where the turning circle would be around the motors, with an omnidirectional wheel to decrease drag and to maintain balance [2].

FU1.3, sensors was designed and built with FU1.4. After thorough investigations, it was found that ultrasonic sensors were the most applicable sensors as they provided accurate measurements regarding the distance of the walls around the mobile robot [3].

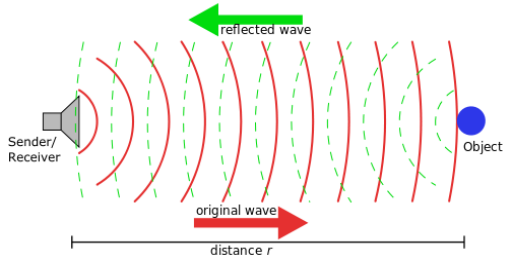


Figure 2: Basic operation of Ultra Sonic sensors

Four sensors were placed on a rotating platform that was controlled by a stepper motor positioned below the platform. The sensors' platform was rotated 90°, pausing every 7.5° to take a distance reading from each of the four sensors to obtain a full 360° view of the walls around the mobile robot. From these distances, the mobile robot was able to determine where to move in the maze and from this information the outline of the maze was displayed, FU1.1.2.

The processing unit, FU1.1, was chosen as a DSPIC33FJ128MC802, as it would allow for the necessary, control over the sensor , motors, display, remote and allow for enough storage eliminating the need for an extra storage device. This processing unit was surface mount and therefore allowed for easy testing.

Integrating the processing system, movement system and sensor system produced the hub of the mobile robot FU1. The mobile robot moved around a wall of the maze, either to the left, right or forward when there is no wall, based on the information from the sensor system. The mobile robot needs to be able to start anywhere in the maze, so an algorithm was used to keep track of where the mobile robot had traveled and where it should travel next [4]. The robot used the information from the sensors to determine the wall configurations and possible routes around them. The summary of the algorithm, FU1.1.2, to control the movement of the mobile robot through the maze is shown in Figure 3 below:

```
If route available to left.
        Move into block on the left.
Else if route available forward
        Move into block in front
Else if route available to right.
        Move into block on right
Else if route available backward.
        Turn around and move into
        block behind
Track movements and orientation
```

Figure 3: Simple sensor operation

The mobile robot tracked where it had moved through the maze in order to not repeat routes unnecessarily and notes the exits of the maze to be used with the shortest route algorithm. The mobile robot then prioritized new movements, after recognizing routes that it has run before. With each movement to a new block, the mobile robot ran a second algorithm to move around the complex walls of the maze. Here, it ran left, right, forward or turned around to run back around the complex walls to move into the new block.
This algorithm allowed the sensor system to obtain the detail of the walls around the mobile robot to determine the movement required to travel around the walls. This algorithm moves the mobile robot half a block each time. This increased the accuracy of the movement, as more data was used to determine the movements and to increase the detail of the display.

The mobile robot was designed to be compact with the movement and sensor system which created an area of 63cm^2 . The maze was thus made to be 1.12m^2. The maze consists of 16 blocks, in a 4 x 4 formation, where each block is 28cm x 28cm. This allowed a mobile robot of the given size to move around the complex options of the maze. The maze layout was as follows:

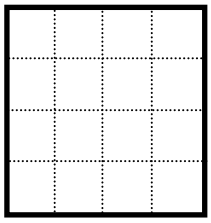
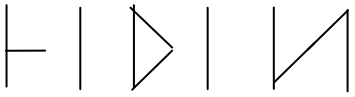


Figure 4: Maze block configuration

The maze allowed for one entrance and one exit to the maze which were 28cm in length. The complexity of the maze came from the wall configurations. The following were the walls available to the maze:



5a) 5b) 5c)

Figure 5: Wall configurations

These walls were used alone or in combinations to create the complex maze.

The display, FU3 of the project was made from the processed information relating to the distances of the walls around the mobile robot. Therefore the positon and orientation of the mobile robot needed to be monitored to normalise the distances for display. The sensor system took a total of 48 distances around the mobile robot for each scan, twice in each block. The distances were then converted into coordinates using the angle that the distances were taken at. The following shows the equation used to calculate the coordinated from each distance reading:

$$(x; y) = (distance * \cos\theta\,;\ distance * \sin\theta)$$

Equation 1: Coordinate system

The coordinates were normalised to the (x , y) positions and orientations of the mobile robot in the maze relative to the starting positon, were tracked by the mobile robot. The coordinates were also scaled and normalised to the pop up screen coordinate system.

The coordinates were sent via UART (Universal Asynchronous Receiver Transmitter) from the DSPIC(Universal Asynchronous Receiver Transmitter) to a PC, via an RS232 to USB (Universal Serial Bus) cable, FU1.3 where the program Putty was used to read the data into a text file. The text file was input into Dev-C++ and the coordinates were plotted as small circles using the “graphics.h” library. The coordinates showed the layout of the walls in the maze in a 2D format.

The shortest route algorithm, FU1.1.3, used the information from the movement of the mobile robot through the maze, and the noted exits. The algorithm would run through the movements of the mobile robot from the entrance and keep track of the distance covered. If a dead end was reached the algorithm ran back until a new route was possible and the distance was adjusted to the new route [4]. When the algorithm found the exit, the route and distance covered were saved. This information was sent with the display information to be displayed with the maze. The algorithm calculated the shortest distance from the entrance to the exit. The algorithm is similar to the algorithm in Figure 2.

Distances of different routes were compared when there was a loop in the maze. However, if there was no loop, then there will only be one route from the entrance to the exit, where either gap was considered as an entrance or exit. The distance was calculated using the following equation where coordinates of the

movement of the mobile robot were used. This was scaled to the size of the maze.

x-current x position
y- current y position
xprev- previous x position
ypres-previous y position

$$distance = \sqrt{|x - xprev|^2 + |y - yprev|^2}$$

Equation 2: Distance of movement

The remote control, FU2 used a wireless communication device, FU1.2, transmitter and receiver, to send information from the remote to the mobile robot. A DSPIC was used in the remote to interpret the input information from four buttons and one switch. Each button allowed for a different instruction: move forward, or backwards, or turn left, or right with the switch used as an on/off command. When the switch was on and one of the buttons was pressed, the DSPIC converts this input into an ASCII(American Standard Code for Information Interchange) character to be sent via UART through the wireless communication device. Each button had a different ASCII character. The mobile robot received the information through the receiver, which causes an interruption. So the mobile robot responded to the input, depending on the input. Figure 6 shows the algorithm used for the remote and the mobile robot for the remote instructions. The switch controlled the power to the buttons and therefore was not needed in the algorithms

Transmitter:

If button 1 input
Send '0'
Else if button 2 input
send '2'
Else if button 3 input
send '4'
Else if button 4 input
send '6'

Receiver:

If UART interrupt

If input matches button 1 command
Move forward
Else if input matches button 2 command
Move backwards
Else if input matches button 3 command
Move left
Else if input matches button 4 command
Move right

Figure 6: Algorithm for remote control

The remote control was integrated with the sensor system to avoid the mobile robot from driving into walls when being controlled by the user.

III. RESULTS

There are five mission critical specifications that were tested.

- The shortest route must be automatically calculated and demonstrated within 4 minutes after the mobile robot has mapped the maze.
- All the walls must be plotted on the PC with an error of less than15%.
- The mobile robot needs to respond correctly to instructions from the remote control at a maximum time of 1 second.
- The mobile robot must move independently around the walls with a minimum distance of 3 cm from the walls.
- The maze must be at least 10 times larger than the mobile robot.

The first result

The second result shows that the walls of the display were plotted with an error of less than 15% for simple maze configurations. However if there are sharp angles and complex corners, errors will arise in the display. Figure 7, 8 and 9, shows the results of the display when specific maze configurations were used.

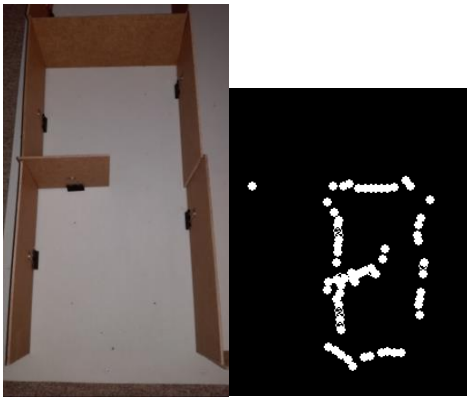


Figure 7: Maze set up and display output

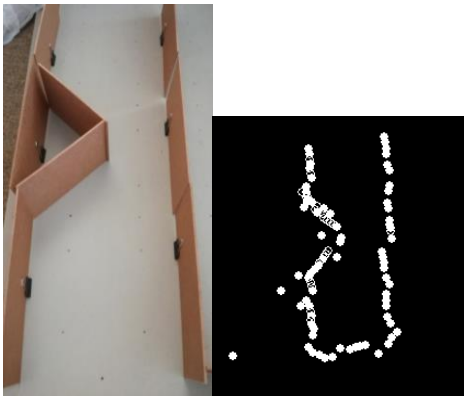


Figure 8: Maze set up and display output

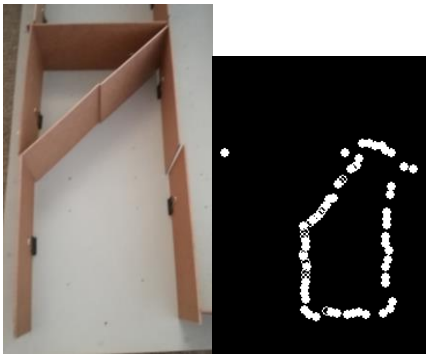


Figure 9: Maze set up and display output

The third result showed that the mobile robot responded to instruction in less than 1s. The theoretical results showed that the information should be sent in under 10ms which is much less than required, and an improvement on the specification.

The fourth result showed that the mobile robot moves around the walls of the maze with a minimum distance of 2cm. This is acceptable as the minimum distance that the ultrasonic sensor can measure is 2cm and therefore will not cause errors in the system.

The fifth result shows that the size of the mobile robot is 6cm x 10.5cm and therefore has an area of 63cm^2. The maze is built so that the mobile robot is able to move around the smallest wall configuration. The maze is built to be 1120cm x 1120cm and therefore has an area of 12544cm^2. The maze is therefore ten times bigger than the mobile robot in length, width and area.

The results from the functional units are summarized below

Specification	Result
The mobile robot	Was less then 20cm x20cm
	Maximum speed was 4cm/s.
	Works wirelessly with in 30cm radius
The remote	Allows for four inputs and a switch to control the inputs
	The height of the walls is less than 15cm, the thickness of the walls is less then 1cm.
The maze	The sensors detect the walls within a distance of 15%, at an angle of less than 30° else they are rejected.

IV. DISCUSSION

Interpretation of results

Sensor system and display

The sensor system was found to work within the required 15% requirement when simple wall configurations are used such as Figure 7 and Figure 8. However the sensors system creates large errors when sharp angles of walls are used as seen in Figure 9. These were found as limitation to the sensor system. To fix these errors a different sensor would need to be implemented such as an image processing sensor system. The sensor system could not read walls that were more than 30°. In order to decrease the errors in the display using the same sensors system. The movements of the mobile robot would need to be smaller, therefore moving four times in each block to gain the detail of the 45° walls and sharp corners.

Movement system

The movement of the mobile robot was tested and the minimum distance of the mobile robot from the walls was 2cm.Which does not meet the specification however the speciation was made for ultrasonic sensors with a range from 3cm. The ultrasonic sensors used in this project have a range from 2cm. The speed of the mobile robot did not meet the 5cm/s requirement because the motors could not provide enough torque for that speed. The maximum speed is 4cm/s. to increase this speed stepper motors with more torque at the same size would need to be used. Therefore the stepper motors would be more expensive.

The maze

The maze was designed from the size of the mobile robot. It was designed to be ten times larger in area, length and width. The mobile robots was kept in the size requirements by more than half in order to reduce the size of the maze to allow the mobile robot to meet time requirements.

Remote control.

The mobile robot responded to the instruction of the remote in a time less than 1second. this increase the control the user has over the movements of the mobile robot which would be required when used with the future car series. The remote control does not work in the full range of the required 10m, to allow for the user to have a large degree of freedom when controlling the mobile robot. The range that the remotes wireless system is 30cm. This is from a low power supply to the wireless communication where they are desired to work on 12V and are used with 5V and 7.4V.

Other aspects to be improved

The movement of the mobile robot runs mainly forward or backwards. If the maze is set up for the most efficient movement through the configuration to be left only. The mobile robot will not move through this efficiently. In order to improve the efficiency of the mobile robot. The code would have been needed to be made longer and more complex in order to normalize the display and movements to four movement instead of two.

V. CONCLUSION

The display of the system was only accurate when wall configurations did not use sharp angles. The mobile robot responded to the instructions of the remote control however the range that the remote worked was much smaller than required. The size of the maze was found to be ten times larger than the mobile robot. The mobile robot moved around the wall configurations with a minimum distance of 2cm. The mobile robots movement system was slower than required as the motors were found to not have the torque to move the mobile robot as fast as required.

From these results the sensor system was not accurate enough for a truly complex maze. The movement of the mobile robot were accurate enough to obtain the detail for the display and the mobile robot was integrated well with the remote control.

REFERENCES

[1] G. Lazaridis, "How stepper motors work," 7 5 2010. [Online]. Available: http://pcbheaven.com/wikipages/How_Stepper_Motors_Work/. . [Accessed 11 10 2016].

[2] R. Ron, "4 wheeled robot design basics and challenges," 27 2 2014. [Online]. Available: http://www.rakeshondal.info/4-Wheel-Drive-Robot-Design. . [Accessed 11 10 2016].

[3] "How does an ultrsoinc sensor work?," GK-12 Program, Comutational Neurobilogy Centre College of Engineering University of Missouri, [Online]. Available: https://www.teachengineering.org/lessons/view/umo_sensorswork_lesson06.. [Accessed 10 10 2016].

[4] A. Dzozdek, Data structures and algorithms in java, Fourth edition.

Part 3. Project identification: approved Project Proposal

This section contains the problem identification in the form of the complete approved Project Proposal, unchanged from the final approved version.

1.1 PROBLEM STATEMENT

Motivation. A Harvard economist, Juliet Schor, explains in her book, *The Overworked American*, that the average worker in the United States worked 160 hours more annually in the 2000s than in the 1960s. This new lifestyle has created the need to save on travel time [1], and thus inspired the motivation for this project: to calculate the shortest route to a destination by using a self-driving mobile robot through a maze. The project will therefore develop a four wheeled mobile robot, as well as an exploration and mapping algorithm that will enable the self-driving mobile robot to find and display the shortest route from a set beginning to a pre-determined destination, in a developed maze, once the entire maze has been explored. The mapping and exploration mobile robot project is part of the future car series main project. The sensor system and shortest route algorithm will later be incorporated with other systems to create the final future series car that will be able to drive automatically though a town or city. The possible benefits from the technology of the future car series include reduction in fuel consumption and emissions due to a reduction in distance and travelling time by finding the shortest route around traffic congestions by avoiding busy routes and accidents zones.

Context. An example of a future car series project is the Google Self-Driving Car. Google have been working on its self-driving car since 2009, but one of the first automatic vehicles was a Mercedes-Benz van, by Ernst Dickmanns, that navigated its way on a highway in the 1980s [2]. The greatest problem encountered by the project was the lack of processing power of computers in the 1980s. In comparison, today the Google car has enough processing power to map its environment and therefore no human interaction is required. In terms of mapping and exploration mobile robot related concepts, the Google Car uses cruise control cameras and has a rotating laser scanner on top of the vehicle to track a 360° span of the environment around the car. The laser scanner allows for digital maps of roads to be created and for the car to locate itself using these maps. A GPS (Global Positioning System) is used for navigation and calculates the shortest route available. The mapping and exploitation project is the start of the mapping technology for South Africa's first future series car. The technology is needed to allow a car to drive and find the shortest route though towns and cities. It will need to be incorporated at a later stage with image-processing technology used to map the road and surroundings of the car [3] [4] and real time traffic data to avoid traffic congestions and accident zones.

Technical challenge. The main technical challenge is to develop an algorithm that will find the shortest route through a complex maze. This challenge creates a need for the design of a complex maze, as well as the design and physical implementation of the self-driving mobile robot that will travel automatically though the complex maze and store a map of the walls without coming into contact with the walls. The stored map will be analysed by the shortest route algorithm, where the mobile robot will then demonstrate the shortest route. The maze and shortest route will be displayed on a PC (Personal Computer) in a 2D (2-Dimensional) map format. Another technical challenge is to design a remote control that will allow the mobile robot to be controlled wirelessly by an external user.

Limitations. The main limitations of the mapping and exploration mobile robot include factors such as the size of the mobile robot, the movement and response time, the battery lifetime, the remote, and the total time for the mobile robot to map the maze. The size of the maze is limited to the size of the mobile robot which should be no more than 10% of the maze. The mobile robot must be able to move and react in real time to the corners and detail of the maze with a four-wheel design without touching the walls. The battery must last long enough to power the mobile robot though the entire maze twice, once to map the maze, and once to demonstrate the calculated route but be sufficiently light to decrease the torque needed from the motors. The remote must be wireless to a maximum of 10m. The whole system will need to run within 10 minutes [5].

1.2. Project requirements

ELO 3: Design part of the project

1.2.1 Mission requirements of the product

Specific details of the system mission requirements are given below.

- The maze must be designed in such a manner so as to allow for alterations to the maze layout. The maze must be complex by having curved walls and 45° angled walls.
- The mobile robot must travel independently through the whole maze by navigating forward, backward or rotating. The mobile robot may not touch the walls when moving through the maze and must track the distance it is travelling.
- The mobile robot must have a second mode, which will allow a user to control the mobile robot with a wireless remote control. However, the mobile robot must automatically stop before colliding into any wall, irrespective of the driving mode.
- The sensors on the mobile robot must read information of the surrounding wall profiles and this information must be processed by the DSP (Digital Signal Processor) and stored on a storage device on the mobile robot.
- Once the whole maze has been mapped, the DSP must run an algorithm to find the shortest route through the maze and this route must be demonstrated by the mobile robot.
- The stored data of the maze must be sent to a PC to create a 2D image of the maze.

1.2.2 Field conditions

The field conditions under which the final product should operate, are:

- The product should operate in the University of Pretoria's laboratory.
- The product should operate under clear sky outdoor conditions.
- The sensors will be adjusted to the lighting and noise level in the labs and ideal outdoor conditions.
- The lab test and mobile robot will be a smaller model to a regular size motor vehicle.

1.2.3 Student tasks

The tasks to be completed in the project are summarised below.

- A maze will be designed and built to be ten times larger than the mobile robot and be changeable if required.
- Suitable sensors, motors, a storage device and DSP will be sourced.
- A battery system will be designed to suite the given mobile robot requirements.
- A mobile robot will be designed, built, and tested to support the wheels, motors, batteries, sensors, circuitry, and DSP.
- An algorithm will be developed and tested to navigate the mobile robot through the whole maze automatically without coming into contact with the walls.

- A method will be developed and tested to process and store data from the sensors.
- A communication system will be sourced and tested to allow the mobile robot to be driven remotely by a user.
- A remote with relevant controls will be created and tested.
- An algorithm will be developed and tested to determine the shortest route.
- A method of transferring data from the storage to a PC will be determined and tested.
- Software will be sourced to allow for the data to be seen in 2D on a PC.
- The software will be used to create a 2D image of the maze on the PC.

ELO 4: Investigative part of the project

1.2.4 Research questions

Research questions will need to be answered in order to complete the tasks and mission requirements of the product. The following are a list of questions to be answered:

- What are the best sensors to accurately determine the distance between the mobile robot and the walls?
- Which are the most efficient motors that are light and powerful?
- Which DSP will be most suitable for the sensors, motors and storage?
- What is the best way to send and receive information to a remote from the mobile robot?
- What is the optimum way to store the information from the sensors?
- What is the optimum way to send the data to a PC?
- Which software will allow for the information to be converted to a 2D map?
- What previous algorithms have been used to find the shortest route?

1.2.5 Student tasks and experiments

The experiments and tasks are given below.

- Test that the final maze on the PC screen is the same as the maze built and that the mobile robot will navigate through the entire maze.
    - Run the mobile robot through the maze.
    - Compare the results on the PC to the built maze.
- Test that the mobile robot can navigate different mazes by changing the design layout of the maze.
    - Change the maze configuration.
    - Rerun and check that the changes have been copied to the PC display.
- Test that the mobile robot does not touch the walls.
    - Have the mobile robot run though many different wall configurations.
    - Check that the mobile robot navigates around all the walls.
- Test that the mobile robot responds correctly to commands from the remote control device.
    - Switch the mobile robot to remote control mode.
    - Push forward button – check that mobile robot moves forward.
    - Turn off remote control access – check that mobile robot returns to

autonomous mode.

- Test that the algorithm establishes the correct shortest route.
    - Find all possible routes to the exit.
    - Measure distance of each of the routes using the mobile robot.
    - Compare the distances of the shortest route with the calculated shortest route from the algorithm.

1.3. Functional analysis

1.3.1 The whole system:

The system is divided into four subsystems - the mobile robot, the maze, the PC display and the remote control. The block diagram of the system is given in Figure 10.

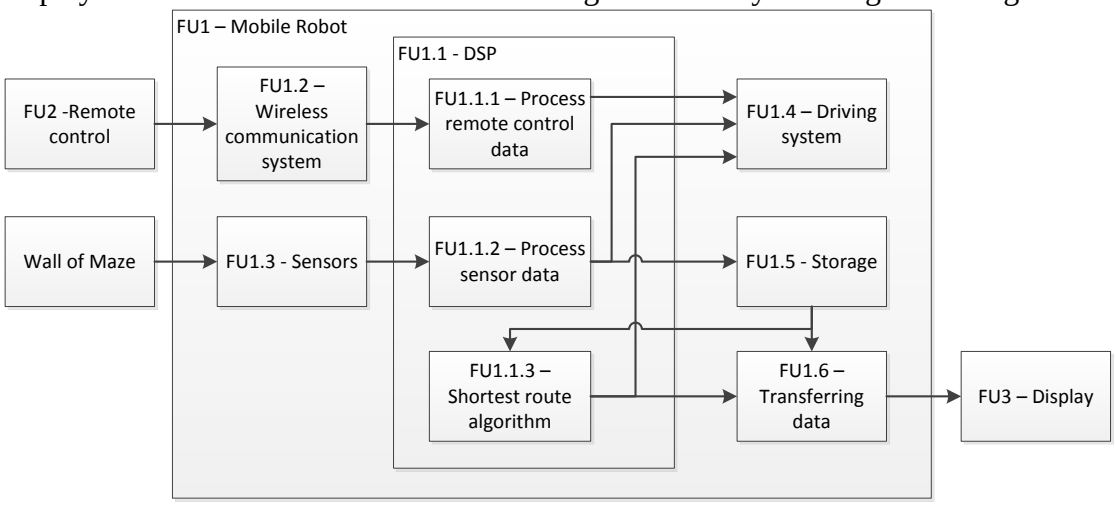


Figure 10: Shows a flow diagram of the whole system

The mobile robot (FU1) will have a wireless communication system, sensors, a processor, a driving system, and data storage. The sensors around the mobile robot (FU1.3) will detect the distance from the mobile robot to the surrounding walls. This distance will be sent to the DSP for processing. The driving system (FU1.4) will respond to instructions from the DSP to move the mobile robot around the maze. A DSP (FU1.1) will be used as it will allow for greater processing power. The DSP will have two data inputs, one from the sensors and one from the remote control. The distances from the sensors (FU1.3) will be processed (FU1.1.2) to send instructions to the driving system (FU1.4). The maze wall information will be processed and stored in external storage (FU1.5). The remote control (FU2) will allow a user to wirelessly send one of five instructions- forward, backward, rotate left, rotate right, and stop- to the mobile robot. The instructions are sent with a wireless communication system (FU1.2) and processed (FU1.1.1) to move the robot accordingly (FU1.4). Once the mobile robot has moved through the whole maze, the data from storage (FU1.5) will be sent to the shortest route algorithm on the DSP (FU1.1.3). The shortest route will be calculated and this data will be sent to the driving system (FU1.4) to demonstrate the shortest route through the maze and sent via a cable to a PC (FU1.6) for display (FU3). The data from the storage is also sent via a cable to a PC (FU1.6) where the maze and shortest route within the maze will be displayed in 2D format, (FU3).

1.3.1.1 The maze

The maze will have the function of being changeable. A set square size maze, as shown in Figure 11, will have set points to add pillars as shown in Figure 12. Each pillar will allow for a wall, to be added at eight 45° points as shown in Figure 13.

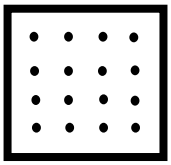


Figure 11: Shows a top view of the maze and where the pivots will be allocated

Figure 12: Shows a top view of a pillar, this shows the slot allocations

Figure 13: Shows the top view of the piece available to create the complex maze

1.3.1.2 FU1.1- DSP algorithm

There are three primary algorithms: one to process the data from the sensors; one to process the inputs from the remote control; and one to calculate the shortest route through the maze. One of the basic algorithms for finding the shortest route is shown in algorithm 3 in Figure 14. This algorithm will be used as the basis to produce the shortest route algorithm specific to the maze. The following flow diagrams show how the three algorithms might work and how they are linked.

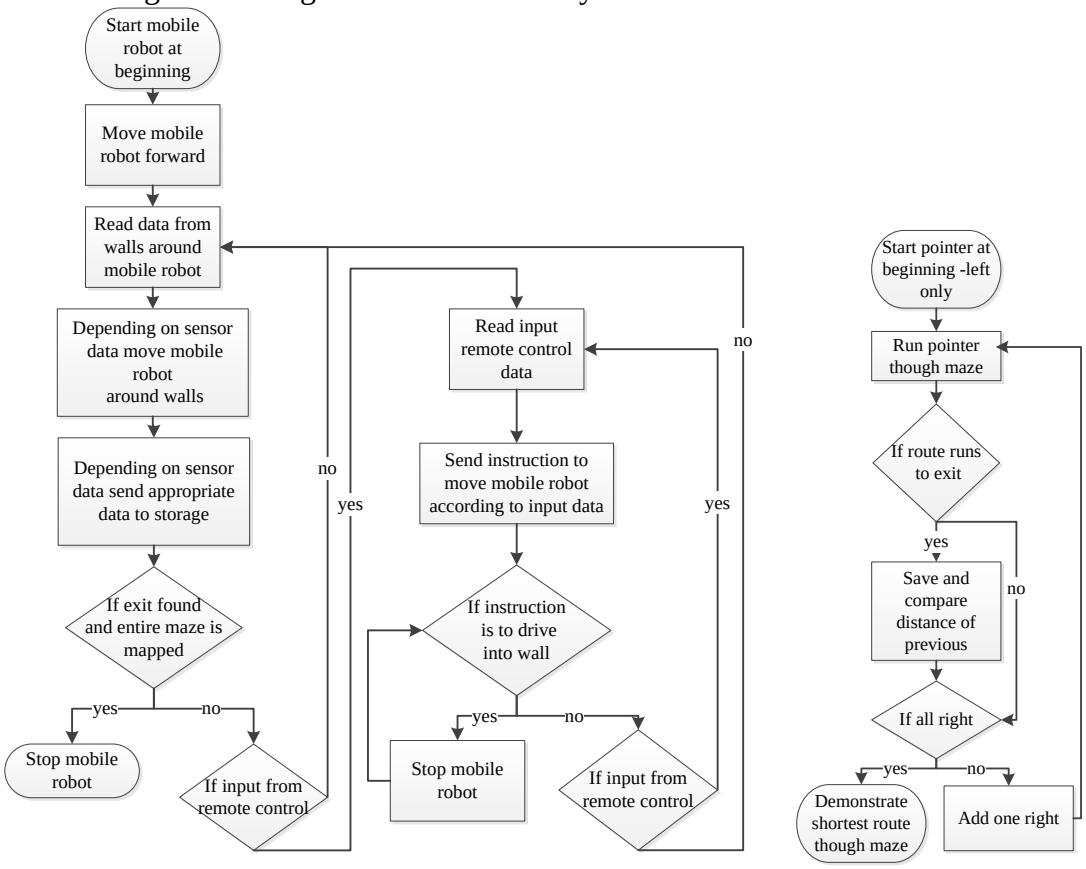


Figure 14: Shows algorithm 1 and 2 on the left and algorithm 3 on the right

1.4. Specifications

1.4.1 Mission-critical system specifications

SPECIFICATION (IN MEASURABLE TERMS)	ORIGIN OR MOTIVATION OF THIS SPECIFICATION	HOW WILL YOU CONFIRM/ MEASURE THAT YOUR SYSTEM COMPLIES WITH THIS SPECIFICATION?
The shortest route must be automatically calculated and demonstrated within 4 minutes after the mobile robot has mapped the maze.	In the National Engineering Robotics Competition (NERC) 2012 [5] the competition gave restriction of 3 minutes maximum to find the shortest route in a 250x350 cm maze. 4 minutes is used for this project as a similar size but a more complex maze is used.	The mobile robot will be placed at the beginning of the maze after the maze has been mapped. A timer will start. The shortest route algorithm will run and the mobile robot will demonstrate the route. The timer will stop on completion of the route.
All the walls must be plotted on the PC with an error of less than15%.	The percent error of distance measurement by ultra-sonic sensors is 11 to 12% [7]. 15% allows for an extra small error leeway for the storage and processing of the sensors information.	The walls in the 2D image on the PC will be compared to the real model by checking that each wall in the real model corresponds to the correct position and orientation in the 2D image of the maze.
The mobile robot needs to respond correctly to instructions from the remote control at a maximum time of 1 second.	A wasp remote control system has delay times of between 100 to 300ms from Tx (Transmission) to Rx (Receive). [8]. 1 second allows for a simple wireless communication system with a slower data rate.	The mobile robot will respond correctly to given external user commands. The response time will be recorded for analysis.
The mobile robot must move independently around the walls with a minimum distance of 3 cm from the walls.	A parallax ping ultrasonic sensor has a minimum distance of 2 cm [7]. Therefore the minimum distance around the walls needs to be greater than 2cm for accuracy.	The mobile robot will move through the maze independently and the distance the mobile robot travels around the walls will be measured with a tape measure.
The maze must be at least 10 times larger than the mobile robot.	This will allow for the car to be small enough to move around different maze configurations.	The area of both mobile robot and maze will be measured with a tape measure.

Table 1. Mission-critical system specifications

1.4.2 Field conditions

SPECIFICATION	ORIGIN OR MOTIVATION

The product should work correctly in a temperature of 5 to 40°C.	Semiconductor parts are specified for use in the commercial range of 0 to 70°C [8]. For example, the operating range of an ultrasonic sensor is 0 to 70°C [6], a DSP is -40 to 105°C [11] and a stepper motor is -10 to 50°C [12]. The product operating temperature is found using these values.
The product will work with indoor lighting conditions (400 to 600 lux).	The product must work in indoor conditions [8].
The product will work with ideal outdoor lighting conditions (10 000 to 100 000 lux).	The product should work under normal external temperature conditions, and not only under controlled lab conditions [8].
The product will work with wind levels of 0 to 7 km/h.	The product must work under low wind speeds, where 7km/h is considered as a light breeze [9].
The product will work with indoor and outdoor background noise conditions of up to 60 dB.	The product must work under normal noise conditions in the laboratories and outside, where 60 dB is considered as conversation in a restaurant, office, background music, Air conditioning unit at 100 feet [10].

Table 2. Field conditions

1.4.3 Functional unit specifications

SPECIFICATION	ORIGIN OR MOTIVATION
FU1. The battery must power the mobile robot through mapping the entire maze twice from the start to the end.	To allow for the mobile robot to map the whole maze and then to power the mobile robot to demonstrate the shortest route.
FU1. The mobile robot dimensions must be less than 20 cm x 20 cm.	To allow for the mobile robot to travel between walls and around the detail of the walls. In the NERC 2012, the competition gave restrictions for the mobile robot to be smaller than 30 x30 cm [5].
FU1.1. The mobile robot must run though entire maze in under 6 minutes.	To allow for the maze to be mapped quickly. In the NERC [5], three minutes is required to find the shortest route, therefore doubling this number will allow for the mobile robot to map the whole maze.
FU1.2. The remote, must be wireless and work within a radius of 10 m.	To allows the user a degree of freedom when controlling the mobile robot. Compared to the WASP remote control system. The system works up to 100 m but this is unnecessary for the product. [9].
FU1.3. The sensors must detect the distance of a wall with an error of less than of 14%.	To be able to move around the walls. The standard percent error of distance measurement in ultra-sonic sensors is 11 to12% [7].
FU1.4. The mobile robot, must drive at a speed of greater than 5 cm/s and slow down to stop.	This speed will allow for the mobile robot to travel though the entire maze within 6 minutes. The maximum speed of a mobile robot in the NERC was 25 cm/s [5] therefore the speed for the mobile robot will be much lower to move around the complex maze.
FU1.4. The mobile robot must turn at a speed of at least 45° in 10 seconds.	This speed will allow for the mobile robot to travel though the entire maze within 6 minutes.
FU1.5. The data storage unit needs to be at least 10 Mb in order to store all the relevant data.	All of the maze data needs to be stored to map the maze on the display. In the report

	"Design and realization of a mobile robot", the processer uses 256 bytes of RAM to do all route calculations. This number has been multiplied by 4 as the maze will be more complex and more detail will need to be stored [9].
FU1.6. The mobile robot must send the stored data to a PC though a cable within 30 seconds.	To allow for the whole system to run within 10 minutes.
FU2. The remote must be battery driven for at least to 20 minutes.	To allow the mobile robot to be driven through the entire maze twice.
FU2. The remote must have five movement options: forward, backward, rotate left, rotate right and stop.	To move the mobile robot in all directions in the maze.
FU3. The display must plot all the walls with an error of less than 15%.	To allow for the shortest route to be shown in an accurate maze. The sensors and processing and storage could add an error of 14% [7].
The height of the walls of the Maze must be less than 15 cm, and the walls must be thinner than 1 cm.	To allow for the mobile robot to travel between walls and around the detail of the walls. In the NERC the walls are a height of 15 cm [5].

Table 3. Functional unit specifications

1.5. Deliverables

1.5.1 Technical deliverables

DELIVERABLE	DESIGNED AND IMPLEMENTED BY STUDENT	OFF-THE-SHELF
Distance sensors		X
Driving system		X
DSP		X
Batteries		X
Algorithm to process remote control input data	X	
Algorithm to process sensor input data	X	
Algorithm to calculate the shortest route	X	
Design of mobile robot to hold sensors, driving system, battery system and DSP	X	
Design of the maze	X	
Remote control	X	
Wireless communication system		X
Storage device		X
Cable for sending data to PC		X
PC software		X

Table 4. Deliverables

1.5.2 Demonstration at the examination

The demonstration will commence with the mobile robot being placed at the beginning of the maze. The mobile robot will run automatically through the maze to show that the mobile robot does not collide with any walls of the maze and how it navigates around the corners and the detail of the maze configuation. Once the mobile robot has completed mapping the maze, the mobile robot will be placed at the beginning of the maze. To demonstrate the shortest route, the shortest route algorithm will again be run on the mobile robot and the calculated route will be demonstrated by the mobile robot navigating through the entire maze. The remote control option will be used to move the mobile robot by an external user to show that the mobile robot responds correctly to user inputs of forward/backward and rotate left/right. The robot will be driven remotely into a wall to demonstrate that the mobile robot stops before colliding with the wall. A cable will be connected to the mobile robot and to a PC where the stored data of the maze walls and shortest route will be sent to the PC. The data will be interpreted and the map of the maze and the shortest route will be displayed in 2D format.

1.6. References

[1] K. Finucan, “Life in the fast lane,” vol. 66, no. 12, p. 10, 2000.

[2] M. Weber, “Where to? A history of the Autonomous vehicles,” 2014. [Online]. Available: http://www.computerhistory.org/atchm/where-to-a-history-of-autonomous-vehicles/. [Accessed 24 March 2016].

[3] B. Templeton, “How might self-driving cars detemine the most efficient route?,” [Online]. Available: www.quora.com/How-might-self-driving-cars-determine-the-most-efficient-route. [Accessed 15 March 2016].

[4] S. Gibbs, “Google's self-driving car: How does it work and when can we drive one?,” 2014. [Online]. Available: http://www.theguardian.com/technology/2014/may/28/google-self-driving-car-how-does-it-work. [Accessed 20 March 2016].

[5] D. M. U. H. Fatima Ahsan, “Seeker: Autonomous Maze-Navigating and,” 2015.

[6] “PING))) Ultrasonic Distance Sensor (#28015),” [Online]. Available: http://docs-europe.electrocomponents.com/webdocs/1382/0900766b81382974.pdf. [Accessed 9 April 2016].

[7] R. solutions, “Secure Remote Control system datasheet.”.

[8] R. Mishra, “The temperature ratings of electronic parts,” Electronics cooling, 1 February 2004. [Online]. Available: http://www.electronics-cooling.com/2004/02/the-temperature-ratings-of-electronic-parts/. [Accessed 14 April 2016].

[9] “STM32F429/439,” ST life augmented, [Online]. Available: http://www.st.com/web/en/catalog/mmc/FM141/SC1169/SS1577/LN1806. [Accessed 14 April 2016].

[10] “Stepper motors, bipolar windings,” [Online]. Available: http://docs-europe.electrocomponents.com/webdocs/13b3/0900766b813b30f6.pdf. [Accessed 14 April 2016].

[11] “The engineering toolbox,” [Online]. Available: http://www.engineeringtoolbox.com/light-level-rooms-d_708.html. [Accessed 10 April 2016].

[12] R. Russell, “Windows to the universe,” NAtion Earth Science Teachers Association, 8 January 2010. [Online]. Available: http://www.windows2universe.org/earth/Atmosphere/wind_speeds.html. [Accessed 10 April 2016].

[13] “Industrail Noise Control,” [Online]. Available: http://www.industrialnoisecontrol.com/comparative-noise-examples.htm. [Accessed 10

April 2016].

Part 4. Main report

1. Literature study

The mapping and exploration mobile robot project requires a mobile robot to be designed and built to automatically navigate through a predetermined complex maze. Once the maze has been fully explored, the mobile robot sends information about the walls via a cable to a personal computer (PC) for two dimensional (2D) display. The mobile robot runs through an algorithm to calculate the shortest route through the maze and then demonstrates the mapped route through the maze. The mobile robot also allows users to control the movements of the mobile robot via a remote control.

The mapping and exploration mobile robot project can be separated into the following sub systems:

- the mobile robot driving system,
- the sensor system that detects and saves the walls details,
- the display of the maze on a PC,
- the shortest route algorithm, and
- the remote control.

Each of these sub systems was researched separately to find the background and context relative to the project. Each subsystem's background context is then linked to this project

I. Background and context

a) Mobile robot driving system

Two separate mobile robots were designed to find the shortest route through a maze by Jano [1]and Ashan [2]. A comparison of the robot properties is shown in Table 5 below:

Properties	Jano	Ashan
Size	Diameter of 40mm	160mm x 220mm area
Motors	Two stepper motors were used to produce a maximum speed of 50 mm/sec	Two servo motors with three wheels, where two of the wheels are used to move and direct the mobile robot and the third is used to stabilize the mobile robot.
Battery system	6.2 V silver oxide batteries were used to power the motors and electronics	An unknown battery system.
Sensors	Photo reflectors were used for sensors, as the mobile robot is a line follower and does not need to map the maze	Infrared (IR) sensors used to detect walls of the maze as the maze is square with little detail and the walls do not need to be mapped.

Processors	The processor has 16 kilo bytes read only memory (ROM), 512 bytes erasable programmable read-only memory (EPROM) and 512 bytes random access memory (RAM) with a power of 600 mW	A controller with unknown specifications.

Table 5: Comparison of robot properties

In the mobile robot mapping and exploration project, the mobile robot was designed to be much smaller and compact, with ultrasonic sensors, servo motors, and Lithium-ion batteries compared to the two articles mentioned in [1] and [2]. In these articles some of the mobile robots use three wheels with two used for steering and movement and the third for balance. A four wheels was considered for this project for navigation and stabilization as stated in the proposal. However a mobile robot with three wheels was the better design and therefore three wheels were used in this project.

Proportional integral and derivative (PID) control has been used to manage a mobile robot's movements around a maze as investigated by Khan [3] , by either keeping the mobile robot on a trace line or keeping the mobile robot in the middle of the maze through maintaining a constant distance away from the walls.

In this project, a form of proportional integral (PI) control was used to correct small errors in the movement of the mobile robot around the complex maze walls using the sensors' distance information. These movements were then stored in memory and used again when calculating the shortest route through the maze.

b) Sensors system

Image processing and cameras have been used as the sensor system for mobile robots as reported by Makhal [4] and Kathe [5], where two cameras are used in order to find the detail and distance of the object in front of the mobile robot. The two cameras capture images (using a stereo vision system) of the robot's surroundings and create a three dimensional (3D) point cloud data using specific image processing techniques. The depth of the object in front of the mobile robot is calculated using the distance between the two cameras on the mobile robot as shown in Figure 15 [4]. In a separate article by Kathe [5], a top view of the maze was captured from above using a universal serial bus (USB) camera. This image is processed and turned into a greyscale format. Image processing techniques are then used to remove light variations and noise. This image is used as an input to Matlab and used to find the shortest route.

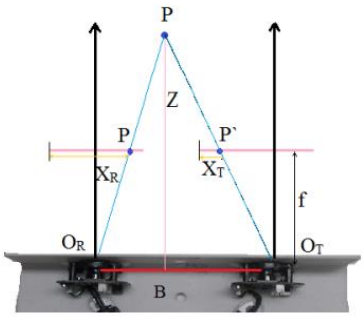
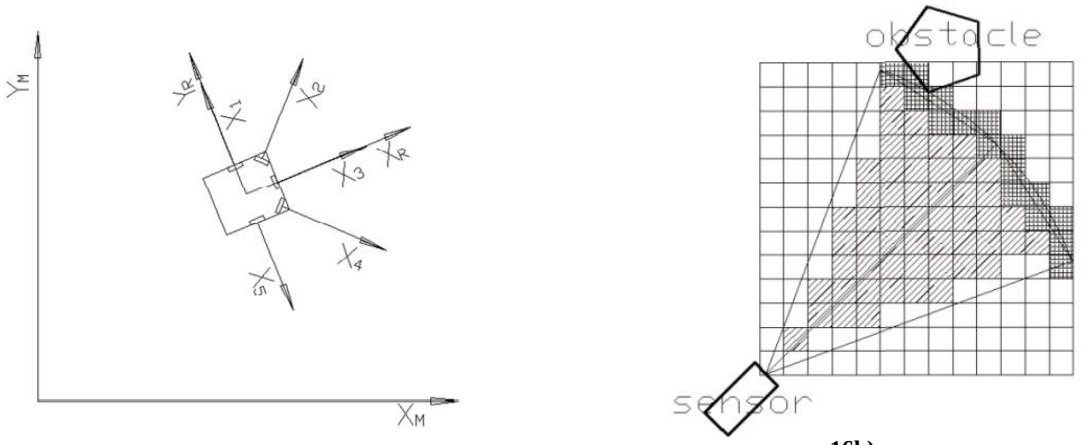


Figure 15: A stereoscopic system with a dual camera system mounted on the mobile robot[4].

It was taken into account that this was a low budget project. So the use of cameras that are exceptionally expensive and add unnecessary costs, as well as the fact that image processing is complex and a project on its own, did not make this option feasable. The use of less-expensive ultrasonic sensors was chosen, which would ultimately obtain a similar required result. IR sensors were found not to give the required detail necessary as compared to the ultrasonic sensors and thus was not considered for this project.

Ultrasonic sensors can be used for object detection and mapping as investigated by Hongyu, Qingxuan, Yili and Kalmegh [6] [7], where the mobile robot has five stationary ultrasonic sensors [6]: one on each side of the robot, one in front and two at 45° from the front as shown in Figure 16a. The mobile robot moves around the object and uses a probability algorithm that makes decisions on the detail and position of the object. Figure 16b shows the beam width of the ultrasonic sensor and how the algorithm would use a grid coordinate configuration so as to determine the position and detail of the object. In a separate article [7], different sensors were considered for the project, which include: bump sensors, IR sensors, and laser range finders. Ultrasonic sensors were chosen as they can determine the range of an object without contact, unlike the sensors from article [7], by sending a pulse and receiving an echo. The distance is calculated from the relationship of the timing difference between sending the pulse to receiving the echo and the speed of sound. The downside is that ultrasonic sensors have a beam angle of 30°, which means that objects at a large angle in front of the ultrasonic sensor will not be detected. To determine the distance of objects around a mobile robot, a single ultrasonic sensor on a stepper motor was used to span 180° [7] and take samples as it moves, thereby receiving more data than many stationary sensors. The amount of detail recorded depends on step size of the motor, where more steps will give more detail.



16a) 16b)

Figure 16: Setup of the five sensors on the mobile robot [16a] and the beam width of the ultrasonic sensor [16b].

In this project, four sensors were used on a rotating platform to gather position information of the walls of the maze. This set up is similar to the set up described in article [7]. More than one sensor was used on the mobile robot to decrease the time of scanning and increase the detail of the walls to be displayed. The mobile robot does not move around an object, but rather the sensors scan ahead of the mobile robot and record the detail of the maze walls. As the walls are in close proximity to the sensors, there was a smaller error from the beam width of the ultrasonic sensors. Saving information of the walls was done using the array of distances obtained from the sensors. These distances were converted into coordinates using the calculated position of the mobile robot in the maze and thereafter plotted on the PC for display. The coordinates were plotted using a computer program. This technique is not often used, as cameras and image processing technology is frequently used to detect and display the detail of the walls. However, using ultrasonic sensors is a low cost compromise.

c) The display

Two articles, [4] and [5], indicate that image processing software in conjunction with cameras were used to display the mazes. In this project, the display uses C code on the PC and the distances of the walls from ultrasonic sensors.

d) Shortest route algorithm

The principles of attraction to free space and the end goal and repulsion to an object around a mobile robot can be used to calculate the shortest route to a set goal destination when a mobile robot is not in a maze [7]. If the robot becomes trapped, a mechanism needs to be deployed to escape from that local minima [7]. The forces are calculated according to Coulombs law in Equation 3:

F=force
Q=charge
R=distance

$$f = \frac{Q}{r^2}$$

Equation 3: Coulombs law.

Therefore the mobile robot moves in the direction of the total force [7], calculated from Equation 4:

$$\vec{F}_{total} = k\sum_{\theta=0}^{2\pi}\left(\frac{1}{t^2} - \frac{1}{|\vec{r}_\theta|^2}\right)\cdot \hat{r}_\theta$$

Equation 4: Total force around the mobile robot.

Two different techniques have been used to find the shortest route through a maze by Yano [1] and Dang [8]. In the first article [1], multiple mobile robots were used in one maze to find the end or goal point. They only communicated when they came into contact with each other. The maze was 2D with start and end goals, and the walls were lines on a paper sheet. A trace line was drawn between the walls, and this is what the robots used to travel through the maze. Therefore, the sensor system needed to detect the trace line and the driving system needed to keep the mobile robot on the line. Intersecting lines were put at the entrances of new paths. These lines were detected by the sensor system and told the mobile robot when a rotation may have been needed as shown in Figure 3. The movement of the robots was based on a state transition map where each state was a different 'obstacle' in the maze, such as a dead end and how to move around these 'obstacles'. When a mobile robot found a dead end, the mobile robot positioned itself at the entrance of the dead end to block it, therefore making the search for the goal point for another mobile robot more efficient. In the second article [8] a mobile robot built for the IEEE micro mouse competition used IR sensors, as the walls of the maze were square with no detail and the walls did not need to be mapped. The maze was 16x16 square cells. A flood fill algorithm assigned values to the cells depending on the distance from the destination or goal cell. The algorithm called for the mobile robot to update the walls, flood the maze, turn determination, and move to the next cell which can be time consuming. A new algorithm proposed that dead ends be eliminated from the calculation process [8]. To determine where the exit to the cells was and the information of the wall, 8 bits were used for each cell. The information was normalised to a set direction such as when the robot was north facing.

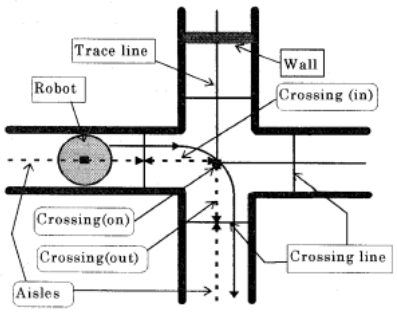
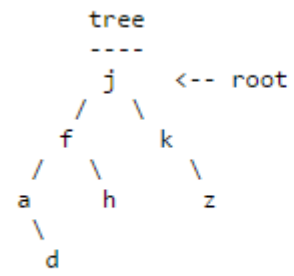


Figure 17: Mobile robot and trace set up of the 2D maze [4].

The depth-first transversal with a tree runs to the deepest part of the tree before back tracking to find a different route. Figure 17 shows how the algorithm works with a tree:



A *preorder traversal* would visit the elements in the order: **j, f, a, d, h, k, z**.

Figure 18: Depth-first algorithm with tree

This algorithm in Figure 18, converted into a maze situation, used the start point of the mobile robot in the maze as the root and therefore creates a tree from the possible routes from this position. [9]

In this project only one mobile robot was used to map the whole maze. The algorithm that calculates the shortest route used the depth-first algorithm as a base. A trace line was not used as the mobile robot needed to map the detail of the walls. The detail of the routes possible, or exits in each block was saved using the distances from the sensor system. These routes and the depth first algorithm are used to calculate the shortest route through the maze.

e) The remote control

The remote control has been kept relatively simple: buttons to control the movement of the mobile robot and a simple, off-the-shelf wireless communication system to send the information to the mobile robot.

2. Design and implementation

.

2.1 APPROACH

The approach to the project was to split the project up into five parts;

- the maze,
- the mobile robot,
- the display,
- the remote control and,
- the shortest route algorithm.

Each of these parts were interdependent on each other's completion, with the shortest route depending on the completion of the maze and feedback from the mobile robot. The remote depends on the completion of the mobile robots movement and sensor system. The display depends on the completion of the maze by the mobile robot. The maze area depends on the mobile robot but the mobile robot does not depend on any of the other parts. Therefore, the mobile robot was designed and built first. In order to test the mobile robot, a smaller version of the maze was built. The display was designed and tested after the mobile robot and small maze had been completed and thereafter the shortest route was incorporated into the coding of the mobile robot. The remote was designed and built last as it was not the main focus of the project.
Figure 19 shows the function unit diagram from Part 3. It will be used to explain the approach from the proposal stage to final product.

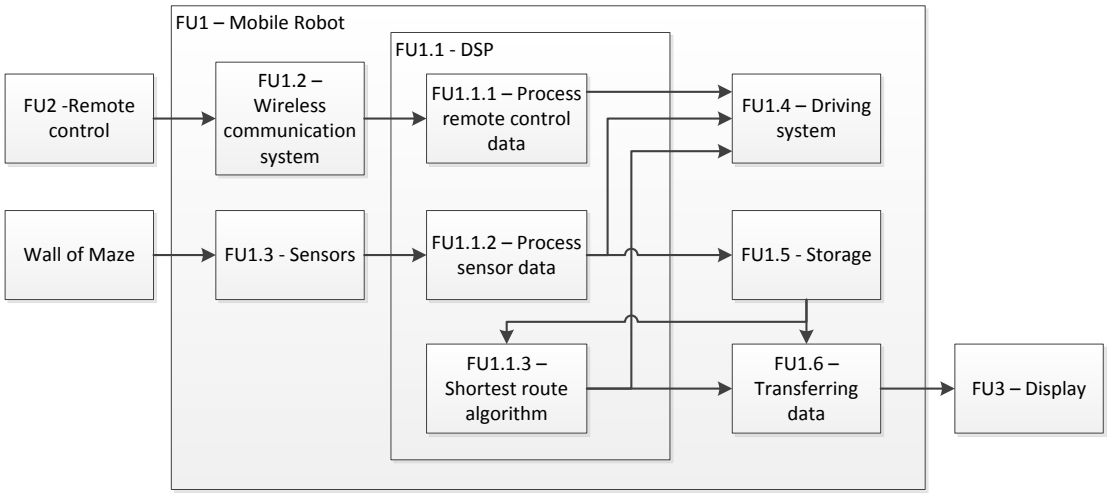


Figure 19: The functional unit diagram

Mobile robot

The design of the mobile robot was created first, FU1. Keeping in mind that the maze size needed to be 10 times larger than the mobile robot, it was decided that the mobile robot was purposely kept compact as a mobile robot with a maximum area of 20cm x 20cm, as originally proposed would result in a maze size of 2m x 2m. The surface area that the mobile robot would need to cover in order to complete the maze would be much larger and this would therefore create a need for a larger battery system. This in turn would then put pressure on the motors and the specification of speed of the

mobile robot, 5cm/s. The specification for time to complete the maze, 6 minutes, would also be put under pressure.

Driving system

The driving system, FU1.4 (within FU1), was designed so that the motors and wheel placement had a small surface area and a tight turning circle to accommodate the structure and complexity of the maze. The motors would need to be small enough and strong enough to move the mobile robot as the mobile robot, was built up, creating more weight on the motors to move the mobile robot, FU1.4. For the motor system, servo motor were originally chosen as they had high torque, but was later changed to stepper motors for an increase in control and therefore accuracy of the movement. The stepper motors allow for precise movements; with 64 steps, 5.625° on each step, increased to 128 steps and 2.8125° on each step, and are small enough to create a tight turning circle by placing the two motors opposite each other and having a third omnidirectional wheel for balance.[10]

Sensor system

The sensor system was designed along with the motors design. The sensor system, FU1.3, needed to give adequate detail of the walls around the mobile robot in order to map the maze. Ultrasonic sensors were chosen as they calculate the distance of an object from the sensor using sound waves where the echo of the sound wave sent is timed to calculate the distance of the walls. Originally five stationary sensors were to be designed and installed on the mobile robot. These sensors allowed an algorithm to make a decision of the proposed configuration of the walls around the mobile robot, and thereby display this configuration, and the movement around it. After careful evaluation this sensor system was rejected as this made the hardware and the software of the mobile robot specific to this one maze (with set wall configurations) and would not work in different maze configurations. The next solution was to use the sensors with a rotating platform driven by a motor to control the platform and thus obtain a 360° layout of the maze walls around the mobile robot. Four sensors were chosen, each rotating 90°, to obtain the detail of the walls around the mobile robot by taking distances every 7°.

Processing system

The processing system was chosen along with the motors and sensors, FU1.1. The processor needed to be able to drive the stepper motors, control the sensor system, allow for communication to a PC and a storage device, and communicate with a wireless communication system. The processor needed to have a large amount of processing power in order to run the movement algorithm, interpretation of the sensor's information for display, and shortest route algorithm. Therefore a DSPIC33FJ128MC802 was chosen. This processor was used to test the motors and sensors.

Motors

The motors were tested to see how they responded to movement commands and the turning circle of the driving system. The servo motors were tested and found to create errors in the movement of the mobile robot that could not be compensated for. For an increase of accuracy in the movement of the servo motors, the motors needed to run slowly, which is undesirable. The stepper motors were tested with different wheel configurations to create the final design with two stepper motors, connected to two

wheels opposite each other with an omnidirectional wheel for balance. This set-up created the accuracy in movement desired with a tight turning circle. An algorithm to control the movement of the motors was created from testing the motors.

Sensors

The sensors were tested to determine the accuracy of the readings relating to the distances and wall angles to ensure they were still accurate. The ultrasonic sensors chosen gave readings that were within the specified 15% error margin for angles of walls below 30°. Therefore, the ultrasonic sensors were not changed. The rotating platform for the four sensors was then designed accordingly. The four sensors were placed on a platform which was supported by a small pole that allowed for the platform to rotate. The rotation is controlled by a stepper motor below the platform, connected to a gear system on the platform. The sensor system with the rotating platform was then tested. An algorithm to control the sensor and store the distances was also created from testing the sensors and an algorithm to control the rotating platform was created. These two algorithms were integrated to save the distances of the walls around the mobile robot.

The two systems were then combined to form the basis of the mobile robot FU1.1, FU1.3, and FU1.4. Combining the algorithms, the motor system needed to respond to an input from the sensor system. Once the mobile robot had the sensor system and driving system combined, the mobile robot was tested in a prototype of the maze.

Maze

The maze was designed to be complex with changeable wall configurations. A set number of wall pieces were then made to be combined in different ways to create the complex maze. The maze pieces are: a full-length wall, half wall, and 45° wall. The maze walls were then attached to a clips that allow the walls to be placed in given slots in the maze. The prototype consisted of 2 blocks by 2 blocks. Each block was made to be 27cm x 27cm. This was back-calculated from the size of the mobile robot and the space the mobile robot needed to move around the complex configurations of the maze. The mobile robot was tested where the mobile robot scanned the walls around it and made the correct decision to move around the walls.

Display

The display was designed once the mobile robot was able to scan the walls around itself. An algorithm was set up to take the saved information of the walls around the mobile robot and convert them into coordinates to be plotted on a PC, FU3 and FU1.1.2. The amount of samples taken was increased to obtain more information about the walls and to plot the walls more accurately. The information needed to be sent to a PC from the DSPIC quickly and easily, so the information was sent to a PC via UART communication, FU1.6. This information was saved in a text file using the program Putty. The information in the text file needed to be displayed in a 2D format on the PC screen, so the program Dev-C++ was chosen to input the data and display the data using the graphics.h library available. This library allowed for the data to be plotted in a 2D format on a pop up screen. The coordinates were plotted as filled in small circles that intersect to form a wall. The PC is used only as an output and therefore the data is not processed in Dev-C++, only plotted.

Algorithms

The movement algorithm was then modified, from moving around simple wall configurations to having the mobile robot being placed anywhere in the maze and finding its way through the whole maze. The movement algorithm needed to know what walls are around the mobile robot and where it is able to move to. The mobile robot uses the information from the sensors to obtain the information needed about the walls for the movement. The mobile robot needs to run through the whole maze from any starting block, so it moves left first, if possible, then forward, then right, then turns around to run backward. The mobile robot also tracks its movements so that it does not run though the same path twice, this tracking system is used to normalise the coordinates for the display. These two algorithms combined allowed the mobile robot to run thought the whole maze from any block. The mobile robot moves half a block each time before rerunning a scan of the walls to increase the accuracy of the display and the movements. The exit and entrance to the maze were noted and used in the shortest route algorithm.

The shortest route algorithm needed to be calculated after the whole maze had been explored. The information of the movements of the mobile robot and the walls of the maze were used to calculate the shortest route from the beginning to the end. The shortest route algorithm is similar to the movement algorithm, where it runs from the one exit moving only left and calculating the distance moved. This route was displayed on the 2D map on the PC.

Remote control

The remote control was designed and built last. The remote needed inputs that will allow for the mobile robot to respond wirelessly to the different inputs; forward, left, right, and backwards, and an on/off switch to control if the remote is on or off. A wireless communication system was bought and tested to send one of four movements to the mobile robot using UART communication. The mobile robot needs to respond quickly to the inputs from the remote therefore a high frequency wireless communication device was used. Push buttons were used as the inputs for the remote and a switch used to control the power. The mobile robot needs to avoid driving into walls when being controlled with a remote control and therefore the sensor system was moved to take reading only at 90° angles to the mobile robot. When the sensor system reads a wall too close in front of it, it stops the mobile robots movement until a left, right, or backwards option is given. Once the remote is used to control the mobile robot the automatic movement is cancelled and will need to be reset.

Hard ware block diagrams

Using the functional unit diagram Figure 20 shows the hardware block diagram:

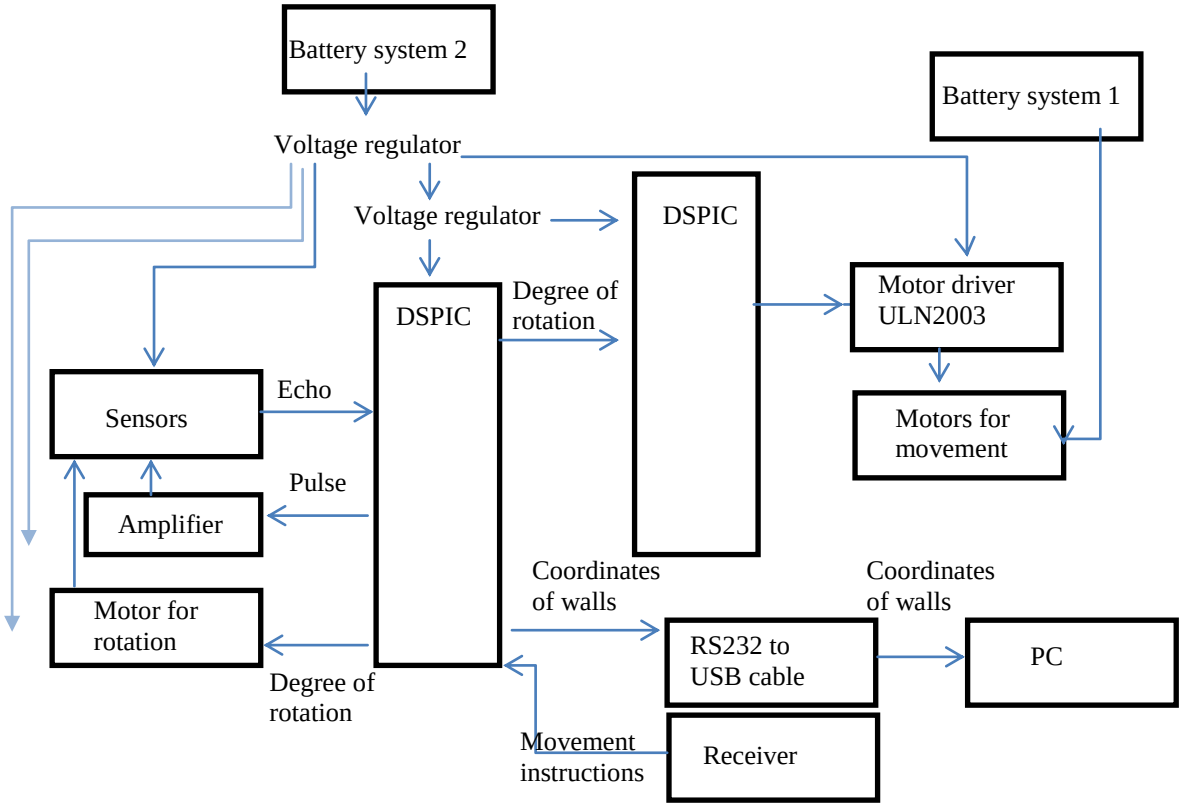


Figure 20: Hardware flow diagram

Figure 21 shows the software flow diagram:

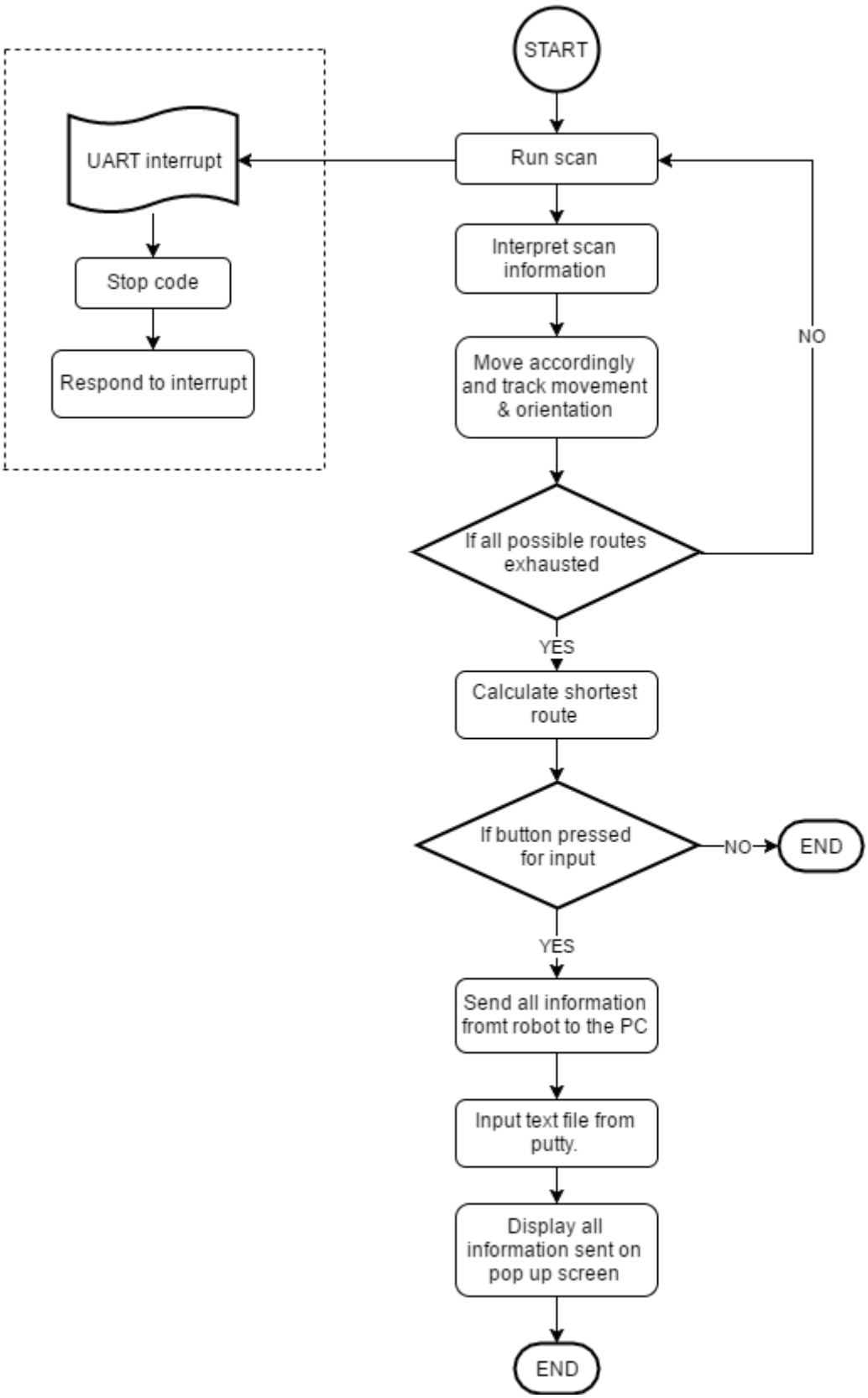


Figure 21: Software flow diagram

2.2 DESIGN AND IMPLEMENTATION

2.2.1 Hardware design

2.2.1.1 Mobile robot

2.2.1.1.1 Theoretical analysis

The mobile robot was designed by separating the mobile robot into three main parts:

- the movement system,
- the sensor system,
- and the processing system.

The mobile robot was planned to be compact and therefore the design has three levels. The first level is the movement system with the motors and wheels. The second level is the processing level with all the processing units and the third level is the sensor level. Each level was set up to be compact and fit on top of each other.
The main challenges with the mobile robot were to optimize the size, movement accuracy, sensor accuracy, and a fast processing system. The mobile robot had to be compact; however, the mobile robot must not be top heavy and create instability. The mobile robot needed to have accurate movements and a small turning circle for errors to not propagate. The distances taken from the sensor system needed to be frequent and accurate for the display and movement. The processing unit needed to be able to handle large amounts of processing for the sensor system, movement algorithm, display, and shortest route algorithm.

2.2.1.1.2 Design alternatives

Servo motors were considered and tested. The servo motors were lightweight, with high torque, and allowed for a compact design were affordable and required low voltage. This allowed for the necessary system on top of the motors to be heavy and the battery system to be lighter and smaller. The servo motors allowed for good speed control with PWM, but not good distance control.
Servo motors run using PWM. A 20Hz square pulse is input into the servo motors and changing the pulse width between 1ms and 1.5ms changes the speed moving in one direction where 1ms is fastest and 1.5ms is stopped. To move in the opposite direction, the pulse width is changed from 2ms to 1.5ms where 2ms is the fastest in the opposite direction and 1.5ms is stopped.
In order to obtain the most control of the distance of the motors, the time that the PWM was sent to the motors was controlled. The slower the motors were running, the more control there was over the movement of the motors. This limited the speed of the mobile robot as more control was needed. Figure 22 shows the input pulses to the servo motors:

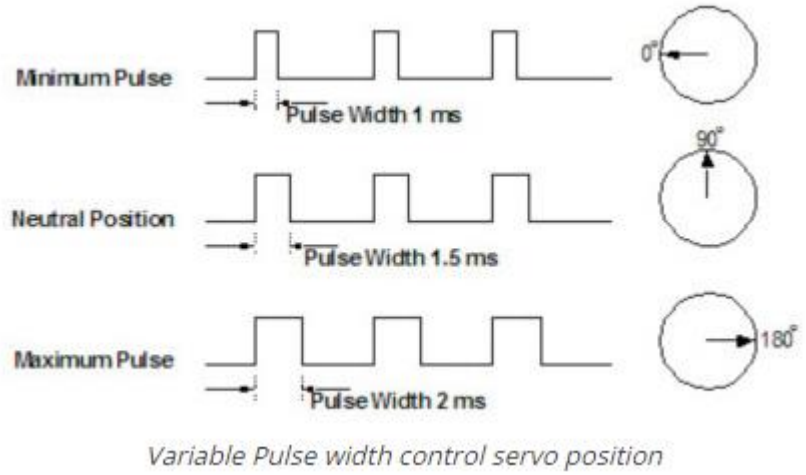


Figure 22: Input pulses to the servo motors [11]

When testing the servo motors with PWM from a DSPIC, each motor reacted slightly differently to the different pulse widths. Therefore, they were calibrated so that they ran at the same speed. This was not ideal as it made it difficult to control and thus created unwanted errors in the movement of the mobile robot. The servo motors could not be placed opposite each other as they would have made the mobile robot too big. So they were placed next to each other as shown in Figure 23 below with the wheels strategically positioned to decrease the drag, the size of the mobile robot, and reduce the turning circle.

.

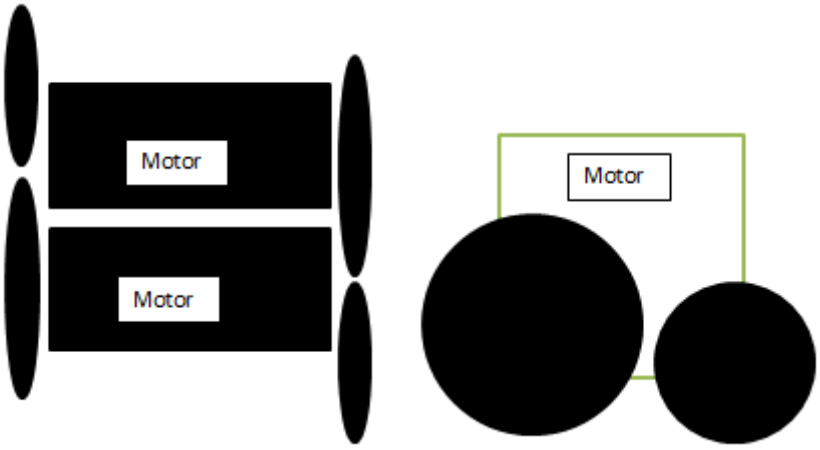


Figure 23: Servo motor design and wheel positions

The servo motors were rejected as they could not give the accuracy of movement that was required. The accuracy was no more than a 5° error in rotation and 5mm error in movement. Small errors would be carried through the maze and became larger errors. The calibration of the motors was not accurate enough and did not give clear results where adjusting the speed of each motor slightly did not always correspond to the adjustment.

A second option considered for the movement system was one where two motors were tested to drive all four wheels. Two wheels would be directly attached to each of

the two motors, these wheels would then be directly driven by the motors, and the other two wheels were alternatively designed to be linked to the wheels being driven using a chain or rubber band system. This would allow for less drag when turning. However, this would not change the problems faced with the servo motors as outlined above.

An alternative design for the sensor system used five stationary sensors. Two levels were to be dedicated to the sensor system with the processing system in between the sensors. Figure 24 shows the planned set up using the five stationary sensors.

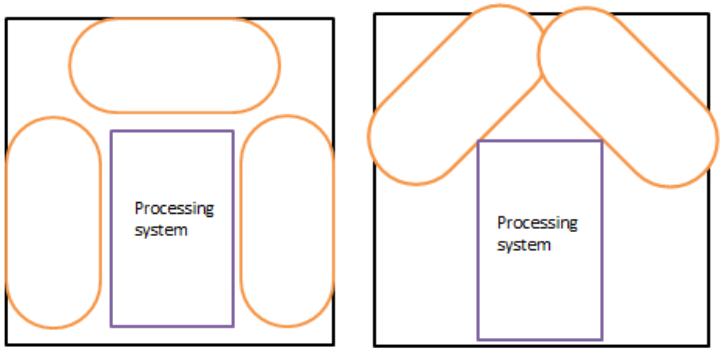


Figure 24: Planned set-up of 5 stationary sensors

This design was rejected as this sensor system was specific to a limited number of wall configurations and did not allow for the mobile robot to be integrated into another maze.

The sensor system chosen for the project has four sensors on a rotating platform. An alternative to monitor where the mobile robot is in the maze, is to add an extra sensor system on top of the designed system or to lift the sensors system up and make the exterior walls higher. The design allowed for the mobile robot to get the distance from its position relative to the exterior walls, and then use this as a reference for the movement and coordinate system. If the sensor system was lifted, this would allow for the movement and coordinate system to be more accurate. However, the sensors chosen do not have a range that would detect the outer walls, so lifting the original sensors would not work. If the sensors were then replaced with sensors that have the range to detect the outer walls, the accuracy of the sensors is lost and the detail of the walls is lost which will create errors in the display. The whole system would run longer as it would need to lift the sensors after scanning the walls for each scan. The other option is to then add sensors on top of the 4 sensors. This would create eight sensors and two DSPICs would need to be used for an increase in number of pins. The mobile robot will draw more current and therefore the batteries will not last as long. The mobile robot would also be top heavy where the system may be unstable with movements. Monitoring the movement of the mobile robot is built into the algorithm for the movement.

Two sensors could have been used as an alternative to four on the rotating platform. Two sensors would be less expensive, but would be time consuming for the mobile robot as it needs to run 180° each time it scans, compared to 90° for the 4 sensor system. The sensors do not use as much power as the motors do to rotate the platform and therefore would not be power efficient. More than four sensors would mean that the system would need an extra level and thus increase the mobile robots height to

keep the mobile robot compact. However, the platform would also lead to a greater weight and the system could become top heavy and unstable when moving.

The sensor system could have been completely replaced with an image processing system, but this is expensive and more complex. Infrared sensors would have been able to detect if there was a wall present but not know how far the wall was which is desired to display the detail of the maze.

The processing unit could have been a PIC or DSP. However a DSPIC has more storage and processing power then a PIC and a DSP is expensive and can be larger than a DSPIC. Most DSPs that were researched needed to be imported and did not allow for a surface mount which is easier to design with.

2.2.1.1.3 Design procedure followed

The movement system was designed first along with the sensor system and processing system. Servo motors were considered, however, after testing, they were rejected and stepper motors where chosen to control the movements of the mobile robot. The stepper motors allowed for controlled small movement and therefore any small errors that were encounter did not become large enough to cause a problem with the accuracy of the movement or display. The chosen stepper motors are 28BYJ-48 12V. These motors gave the torque required, had a stride angle of 5.625°/64 for movement accuracy and were compact in size and less expensive than other stepper motors.
The stepper motors work with magnets where the magnets are excited in an order to rotate the shaft around. In this case, half stepping was used to increase the control of the movement. Exciting an outer metal piece creates a magnet that has a north and south pole, this attracts the shafts magnet's opposite pole, where a south pole attracts a north pole, as seen in the first image in Figure 25. Exciting a second metal attracts the shaft towards the second magnet. However, the shaft is still being attracted to the first coil and therefore stops between them. This increases the steps in the motor through software, and with more steps the degree that the mobile robot moves is controlled more accurately. The following shows how the half stepping works inside a stepper motor:

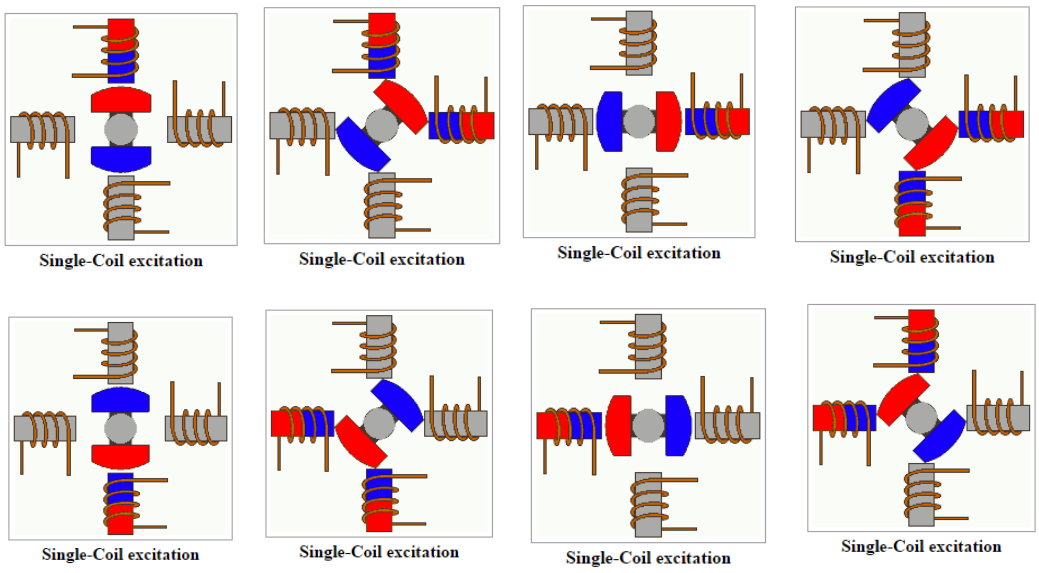


Figure 25: Sequence of half stepping inside a stepper motor[12]

The following shows the code used to control one motor in a clockwise rotation. The delay between sending each code controls the speed of the motors. Where a longer delay produces a slower speed but an increase in torque. This code was sent through a decoder, the ULN2003 where the output of the decoder controls the stepper motors (clockwise).

0001
0011
0010
0110
0100
1100
1000
1001

To turn the stepper motor in the opposite direction,(anticlockwise) the code is run backwards. The following code is sent to the motors:

1001
1000
1100
0100
0110
0010
0011
0001

Because the mobile robot would be moving though a complex maze, a small turning circle would be required to move around the complex walls and to easily keep track of the movement of the mobile robot. Figure 26, 27 and 28 show how a turning circle changes with a change in wheel position using four wheels and two motors.

In Figure 26, the motors are connected to the back wheels. This set up creates a large turning circle and drag on the front wheels. This is not what is desired for the mobile robot in this project as there is an unnecessary loss of torque and battery power.

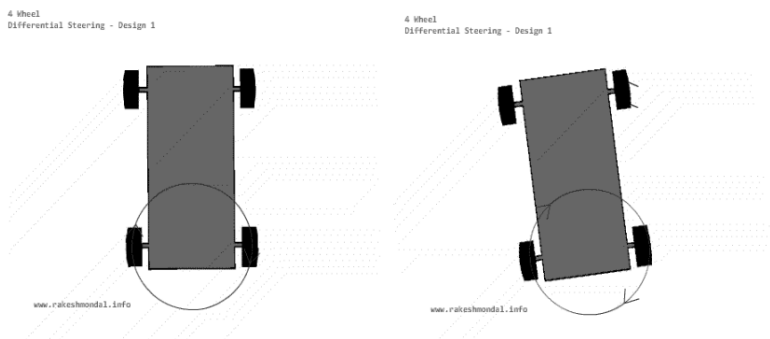


Figure 26: Motor connection to back wheels

Figure 27 shows the front wheels positioned closer to the back wheels. This creates a smaller turning circle however there is still drag on the front wheels.

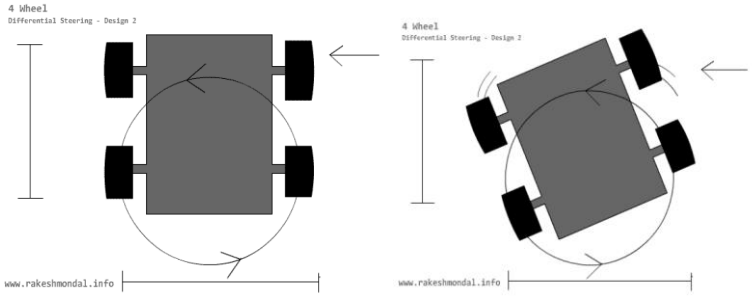


Figure 27: Front wheel position closer to back wheels

In Figure 28 the wheels are close together. This creates a small turning circle with minimal drag on the front wheels. This creates a fast rotating action with a small turning circle and little loss of power.

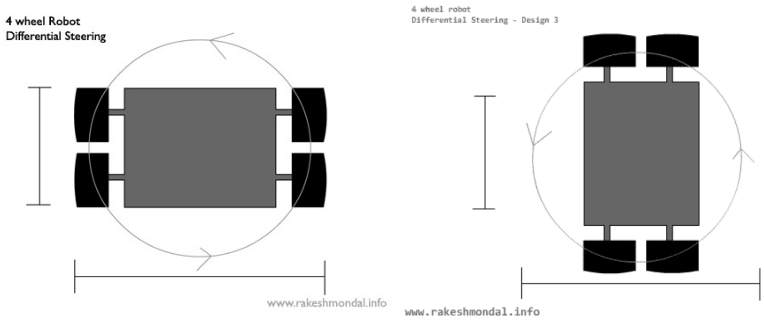


Figure 28: Wheel postion very close together [13]

These designs are optimal when four wheels are used, but, they still create drag on two of the wheels and therefore the turning circle is not ideal. To reduce the drag on the wheels, two motors are placed opposite each other, where the wheels are opposite each other and an omnidirectional wheel is used for balance. The omnidirectional wheel will create the least friction for all movements of the mobile robot. With the wheels opposite each other, the turning circle is minimised around the motors when each motor is driven in the opposite direction to turn the mobile robot. This design was used in the project. Figure 29 shows the motor and wheel set up used for the mobile robot:

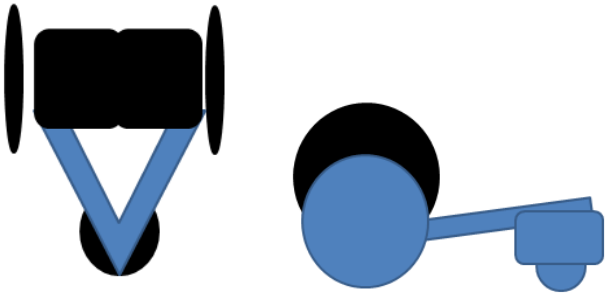


Figure 29: Motor position with omnidirectional field

The method of driving used was differential driving where the movement of the mobile robot depends of the velocity of the wheels. If both motors run in the same direction, the mobile robot runs forward or backward depending on the direction. The mobile robot turns left and right when the motors run in opposite directions. [14]

The sensor system was designed aside the design of the motors. The sensors chosen were US-100 Aduino ultrasonic sensors which are specifically accurate and relatively inexpensive. The ultrasonic sensors chosen had a range between 2cm to 500cm and an accuracy up to 1mm, which would allow for the mobile robot to display the walls that are close to the mobile robot accurately.
The ultrasonic sensors work by sending a short pulse, greater then 10us, and waiting for an echo to reflect back. The time-difference between the sent pulse and the echo is used to find the distance from the sensor to the wall. The pulse that is sent to the ultrasonic sensors, however, needs to be 5V therefore the output from the DSPIC is sent though an amplifier circuit before being sent to the ultrasonic sensors.
Figure 25 shows the sound waves created and reflected back from an object:

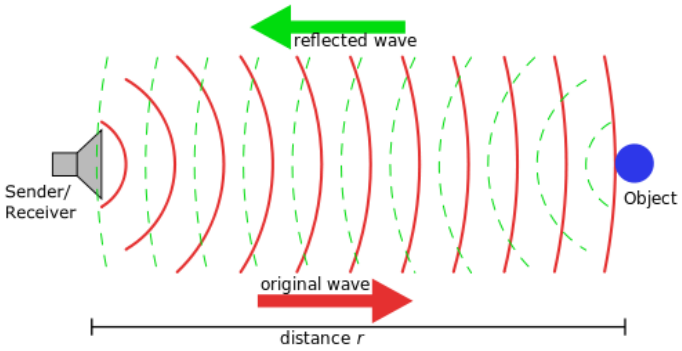


Figure 30: Sound wave patterns from an object [15]

The system to rotate the sensors for a full 360° scan of the walls around the mobile robot was developed after the sensors had been tested. Four sensors were placed on a platform, which allowed for rotation at 90° facing outwards. The set-up is shown in Figure 31:

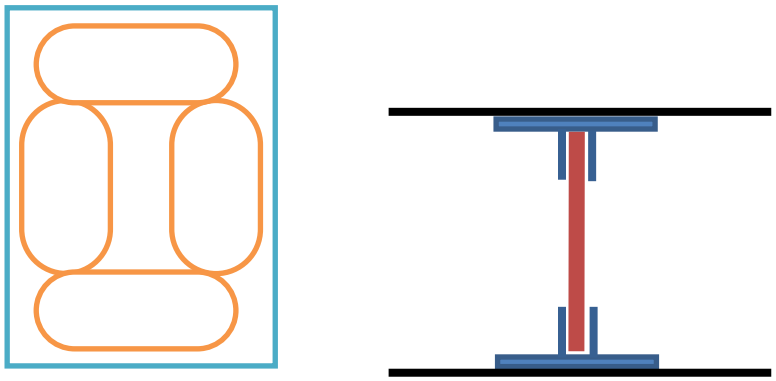


Figure 31: Sensor rotation platform

The red pole is the pivot pole that allows the top platform to be rotated regardless of the rotation direction of the bottom platform. The top platform had a gear system with teeth facing down. This allowed for a stepper motor attached to a gear to control the movement of the top platform. Figure 32 shows the gear system used to control the platform:

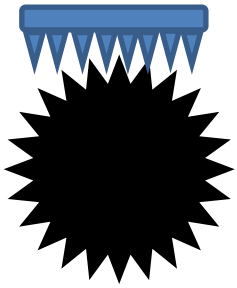


Figure 32: Sensor rotation platform [16]

Since the rotation of the platform needed to be smooth with little friction to control the movement accurately and reduce the strain on the motor, ordinary wires controlling the sensors from the processing system below are too thick and do not bend or move easily. Therefore, thin, grey wires were used as they moved easily with the movement of the platform. Figure 33 shows how the platform, gear system and wires are used.

Figure 33 shows the sensor system.

The more teeth in the gear system, and the smaller the stepper angle of the motor, the more control of movement. The side gear was controlled by a stepper motor that runs the gears clockwise and anticlockwise to rotate, and rotate back the platform. The degree that the platform moved depended on the amount of steps taken by the stepper motor. The sensors and rotating platform were integrated to form the final sensor system. The sensors each take two distance readings and average the distances. The distances are saved in an array and converted into coordinates depending on the angle of the platform. The converted distances are saved in another array. The platform moves every 7.5° and the sensors calculate new distances. These distances are saved into an array and converted into coordinates with a new angle. This system runs twelve times over the 90° of movement, each time saving the distances and converting

them into coordinates. A total of 48 distances around the mobile robot are saved and converted into coordinates.

The processing unit was chosen along with the sensor and motor system. The processing unit chosen was a DSPIC33FJ128MC802 as it allowed for a large amount of memory, it was affordable, fast, allowed for PWM for motor control with the original servo motors. The DSPIC has enough storage on board and therefore an additional storage system was not necessary. The DSPIC has up to 40MIPS 16-bit sdPIC33f CPU. Four PWM generators for the servo motors that were replaced. Two UART communication interfaces for the display and remote. I^2C if extra storage was required and 16Mbytes of RAM for storage.

Combining the movement system, sensor system, and processing system created the mobile robot. The three systems were integrated through the processing unit's algorithms, where the movement system depended on the input from the sensor system.

The battery system was designed last. The batteries chosen are 3.7V 1500mAh. These batteries are small, rechargeable batteries with a large amount of mAh to drive the motors for longer periods of time. The two batteries created 7.4V which was put through a voltage regulator to create 5.5V. This was used for the amplifier, the stepper motor for the sensor system, and the sensors. This voltage was then put through another regulator that reduces the voltage to 3.4V for the DSPIC. The voltage regulators did not waste as much power as voltage divider circuits and were therefore chosen. Three separate batteries, 11.1V, were used to run the movement system with the stepper motors. This allowed for less interference in current and voltage between the two systems and power to be directed to the movement system. The batteries were placed between the movement system and processing and sensor system.

The amount of pins on the DSPIC were chosen to work with servo motors as initially designed where two pins were required to control the two servo motors. However, when stepper motors are used for the movement system, eight pins are needed. Therefore a second DSPIC was used where four pins from the original DSPIC had been used to send information of the movement and distance: forward, backward, or turning, to the second DSPIC where it interprets the information and drives the motors accordingly with eight pins. The movement of the motors is controlled from each pin, where each pin is either a forward, backward, turn left or turn right command. The distance that the mobile robot needs to move is controlled by the length of the command signal. The distance that the mobile robot needs to move is controlled by the length of the command signal

2.2.1.1.4 Design equations and design calculations

Amplifier design equations:
The IC LM224N is an operational amplifier that allows for 5V and ground to be input as the power.
The following shows the design equations used to increase the voltage from 3.4V to 5.5V

V_o – voltage out
V_i – voltage in
A - gain

$$Vo = (A)Vi$$

$$A = \frac{R1 + Rf}{R1}$$

$$A = \frac{12000 + 12000}{12000}$$

$$A = 2$$

Figure 34 shows the circuit diagram of the amplifier to increase the voltage from 3V to 5V

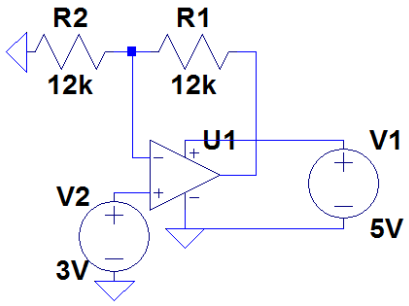


Figure 34: Circuit diagram of the amplifier

Figure 35 shows the output of the amplifier circuit.

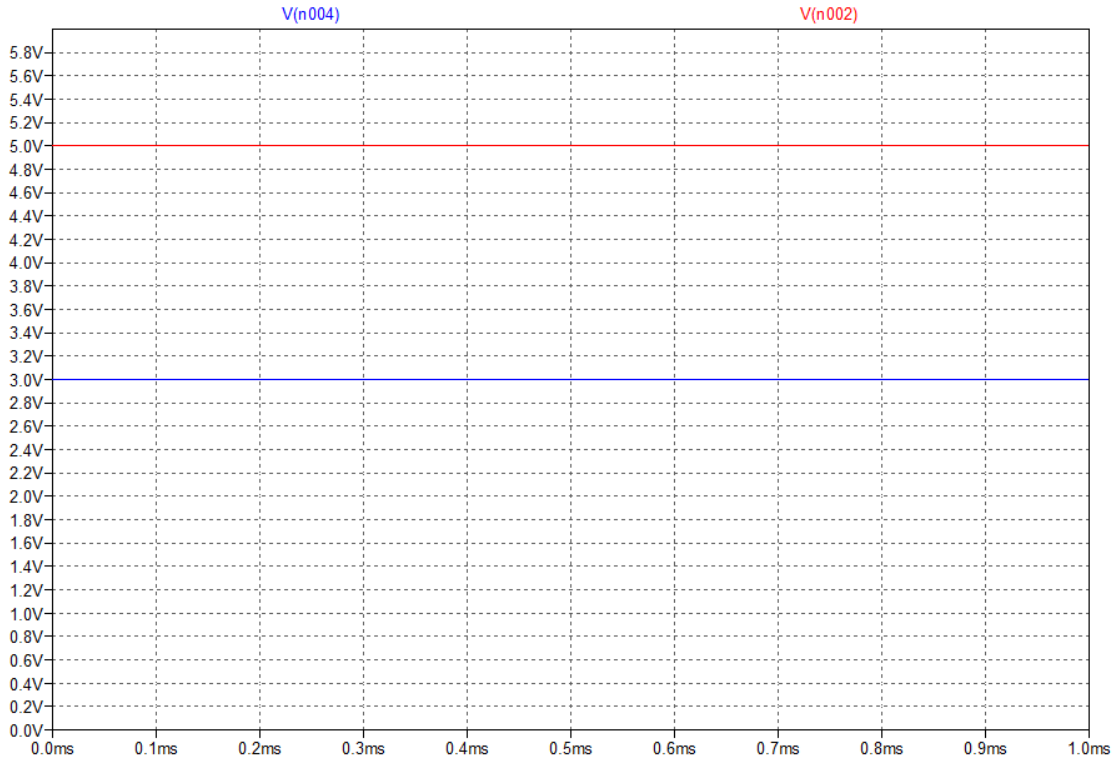


Figure 35: Ouput and input voltage of the amplifier circuit

Voltage regulator design equations:

The IC LM317 is a voltage regulator where the output is kept at a specific set voltage within a range from the input voltage. Two voltage regulators were used, one for the input voltage for the sensors, amplifier and stepper motor and the second for the DSPICs.

The input voltage is 3.7V+3.7V = 7.4V which needs to be brought down to 5.5V for the amplifier, sensors and stepper motor. Figure 36 shows the voltage regulator used:

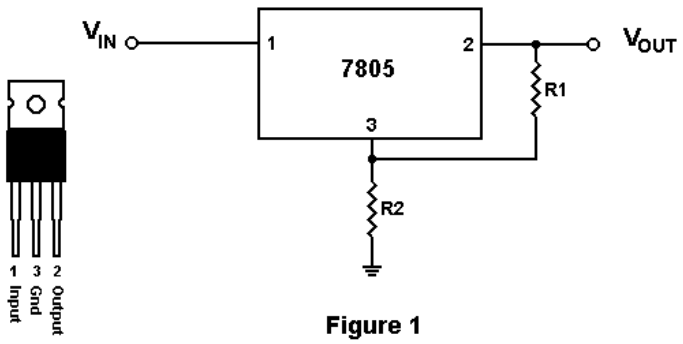


Figure 36: Voltage regulator circuit[17]

the following shows the design equations used for the voltage regulator:

V_{out} – voltage out

V_{ref} – voltage reference

$$Vout = Vref\left(1 + \frac{R2}{R1}\right)$$

$$R1 = 270$$

$$3.4 = 1.25\left(1 + \frac{R2}{270}\right)$$

$$R2 = 470$$

$$Vout = Vref\left(1 + \frac{R2}{R1}\right)$$

$$R1 = 270$$

$$5.5 = 1.25\left(1 + \frac{R2}{270}\right)$$

$$R2 = 918$$

2.2.1.1.5 Final design schematics

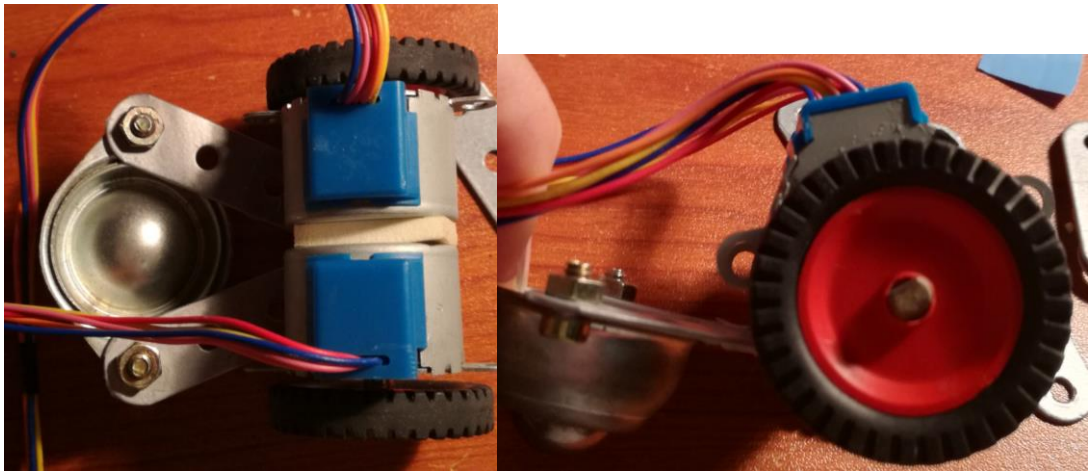


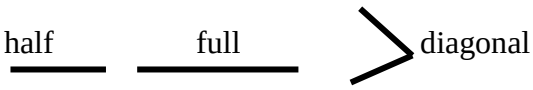
Figure 37: The motor and wheel configuation

2.2.1.2 Maze

2.2.1.2.1 Theoretical analysis and design calculations

The complex maze was designed before the mobile robot was built to ensure that the mobile robot was able to handle the different complex maze options. The maze was then scaled to the size of the mobile robot afterwards. The complex maze was chosen to have a square exterior frame where the level of complexity comes from the walls inside the maze.

The square maze and is split up into 16 blocks (4x4). Each block can have a combination of walls where the choices of walls are; a half wall, a full wall and a diagonal wall.



Because the maze is complex, restrictions needed to be set as to how complex the maze will be. The restrictions chosen: there are only two exits/entrances to the maze, they are a full width (across the whole block), and each block is the same size. The maze was set up to allow for the set number of complex combinations using the given walls available. The mobile robot needed to start in a simple block, which is defined by having no walls inside the block.
Once the mobile robot was built the maze could be designed and built according to the size of the mobile robot. The smallest width that the mobile robot will need to move around is shown in figure 38:

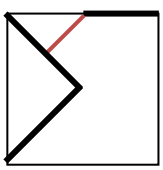


Figure 38: Smallest width of path

The red line indicates the shortest distance that the mobile robot will need to move through.
If the blocks' dimensions are 10cm x 10cm then the distance of the red line is calculated as follows:

$$r = \sqrt{x^2 + y^2}$$

Equation 5: Pythagorous Rule

$$= \sqrt{5^2 x 5^2}$$
$$= 7.02$$

$$\frac{7.02}{2} = 3.51$$

If the mobile robot had a width of 6cm, adding 2 cm each side for movement, the mobile robot could move through a maze that is 28cm x 28cm.
By using 28cm x 28cm for each block, this made the total maze 1.12m x 1.12m. The size of the mobile robot is 6cm x 10.5cm and therefore the size of the maze is ten times larger than the mobile robot as required.

2.2.1.2.2 Design alternatives

The design alternative for the maze walls placements was a pillar system where pillars were set up in the maze and walls are slotted into these pillars. The pillars would allow for walls to be placed at 45° and 90° angles. The size of the pillars would be large and interfere with the movement of the mobile robot around the maze. If the size was decreased, there would be less support to hold up the walls and there were no easy materials to create the 45° pillars. The pillar design was therefore rejected.

The outer walls may have been needed to be made higher if the sensors' system was used to work with the higher outer walls. The sensor system was rejected, and therefore the maze was not required to have higher outer walls.

2.2.1.2.3 Design procedure followed

A prototype test maze was first built in order to test the mobile robots movement algorithm and display. The size of the test maze was 2 blocks by 2 blocks where each block is 27cm x 27cm. The material used to build the maze was hardboard. It is strong and thin and can be cut into any different maze walls needed. A hardboard base was cut to support the walls and act as a floor for the maze. The base was reinforced with a pinewood frame to give rigidity and prevent warping. Holes were drilled into the base at predetermined points and bull dog clips are securely screwed onto the base.

The hardboard walls were slotted into the bull dog clips to hold them upright. The holes were small enough to not interfere with the movement of the mobile robot if they were unused.

Figure 39 shows how the clips were attached to the base. A screw runs through the base of the clip and through the board. The screw is held in place with a bolt. The length of the screw needed to be less than the thickness of the wood frame around the base.

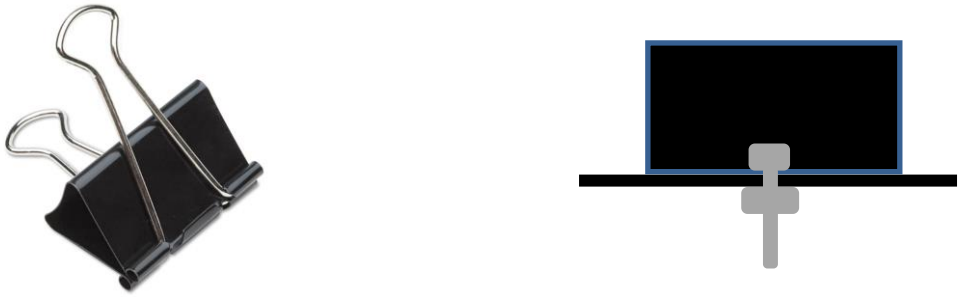


Figure 39: Bull clips for wall support [18]

2.2.1.2.5 Final design schematics

Figure 40 shows the final design schematics of the maze.

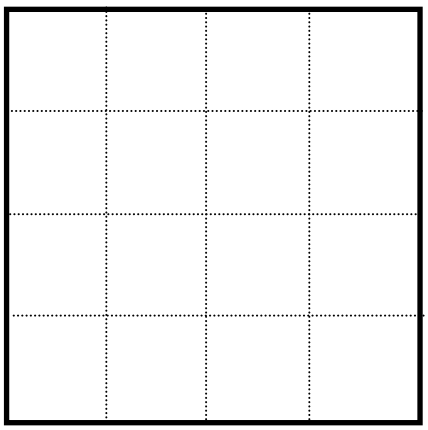


Figure 40: Final maze design showing block arrangements

Figure 41 shows the building schematics for the maze. The maze base was split into two parts to allow for easy transport. The two bases were split as shown by the green line in Figure 41. The two bases were given a frame to strengthen the boards and keep them flat. The 16 blocks were then placed on the two bases shown below in the orange.

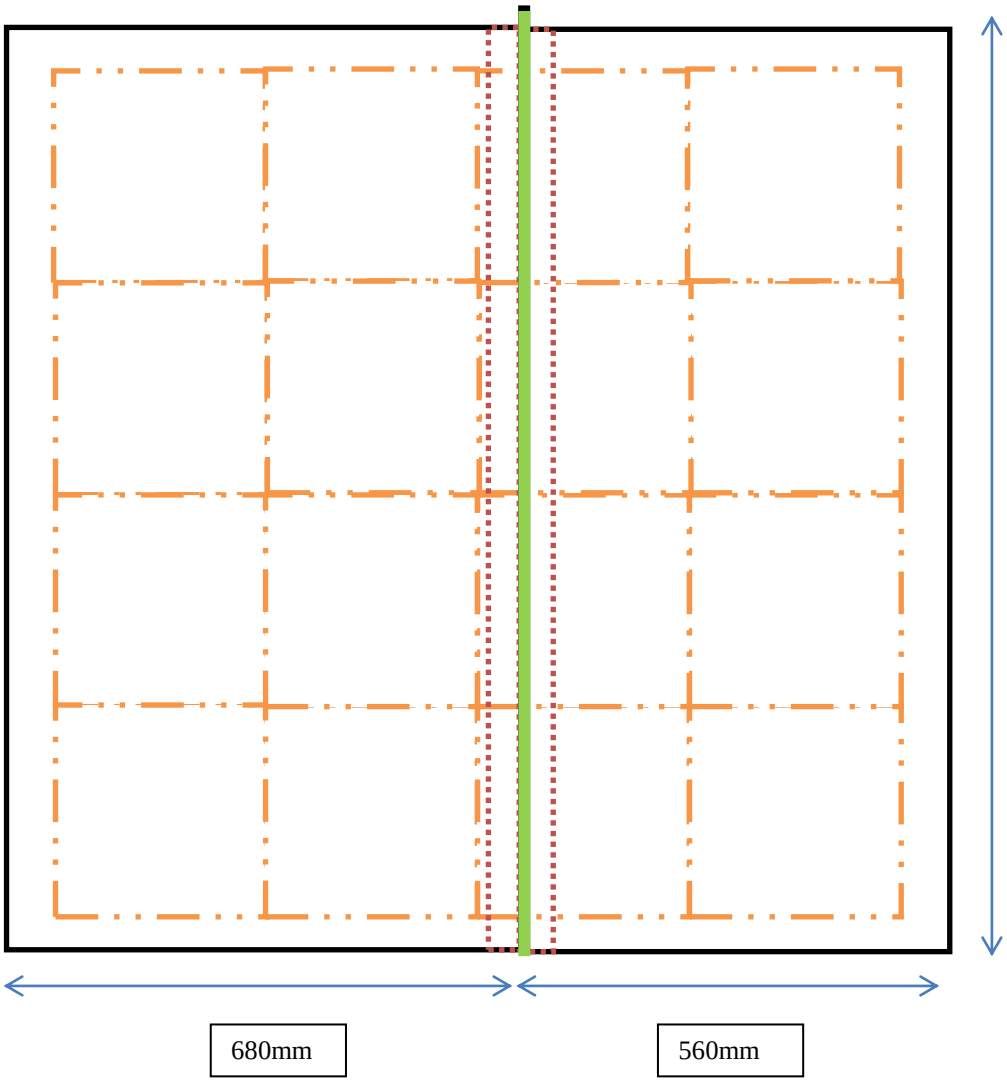


Figure 41: Sensor rotation platform

Figure 42 shows the hole placement in the inner block of the maze. This allowed for the installation of half walls and 45° walls in many combinations.

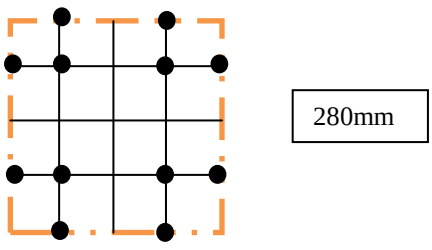


Figure 42: Sensor rotation platform

2.2.1.3 Remote Control

2.2.1.3.1 Theoretical analysis

The remote control was designed to be simple. The remote allowed for five inputs; forward, rotate left, rotate right, backwards, (controlled with simple push buttons) and an on/off switch. The stop signal was activated when any of the push buttons were released. The remote sent the relevant information to the mobile robot for processing and interpretation.

2.2.1.3.3 Design procedure followed

The store-bought wireless communication device was tested by sending ASCII characters through UART communications. The wireless communication device consisted of a transmitter and receiver. The binary ASCII information was sent through the transmitter and amplitude modulation specifically ASK(amplitude shift keying) was used to convert the binary signal into a cosine signal that can be sent to the reciever. The carrier amplitude, $\cos\omega_c t$, is varied in proportion to the modulating signal where the frequency is 433MHz. Figure 43 how the binary signal is modulated using ASK.

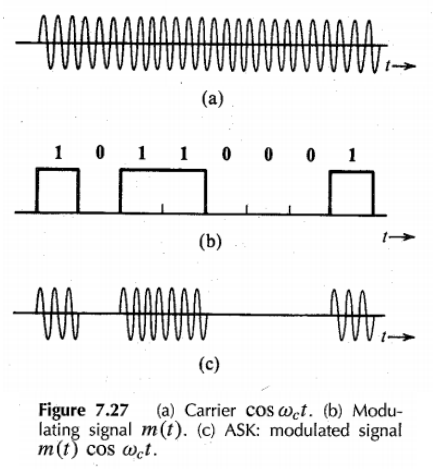


Figure 7.27 (a) Carrier $\cos \omega_c t$. (b) Modulating signal $m(t)$. (c) ASK: modulated signal $m(t) \cos \omega_c t$.

Figure 43: ASK modulation[19] [1]

Once information could be sent from the transmitter to the receiver, the input buttons were linked to the communication system where the input from each of the buttons sent a specific ASCII code that was picked up on the mobile robot.

The processor used for the remote is the same processor as the mobile robot, the DSPIC33FJ128MC802. This processor was easier to use as it had been set up and tested with the mobile robot.

Each of the buttons was linked to an LED to indicate when they were being pressed. The switch was linked to an LED to show when the remote was on or not. The switch controlled the voltage over the buttons. When the switch was off, no voltage ran through the buttons and the therefore the LED was off. When the switch was on, voltage ran to the buttons and through the buttons, as well as the LED when the buttons were pressed. Only one command could be read at a time. Figure 44 shows the setup of the buttons and LEDs.

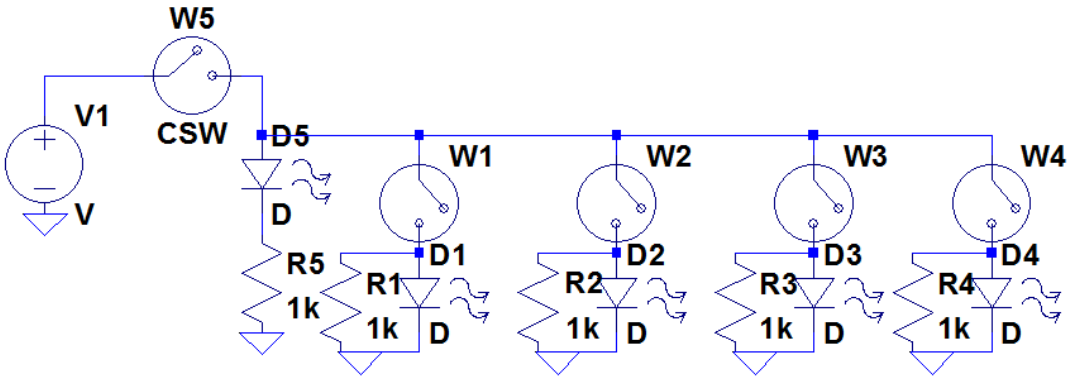


Figure 44: Sensor rotation platform

The mobile robot received the ASCII characters as an interrupt where it stopped the automatic code for the maze and responded to the commands from the remote push buttons. Once the mobile robot responded to the remote control instruction, it needed to be reset if desired to run through the maze. The remote allowed the user to control the mobile robot through the maze; however, it would not map the maze and therefore will not calculate the shortest route.

The mobile robot's sensor system was linked to the remote control instructions. The sensor system repositioned so that the front sensor was parallel with the mobile robot. As the mobile robot responded to the inputs from the remote control, the sensors ran to check the distance of the wall around the mobile robot. If the mobile robot was being driven directly into a wall, using the remote control, the mobile robot would stop and wait for a command that would move it away from the wall.

The battery system used two 3.7V batteries to power the communication device. This voltage was put through a voltage regulator to reduce the voltage to 3.4V for the DSPIC, buttons, and LEDs. The higher the voltage for the communication device, the further the range the wireless communication system would work.

2.2.1.3.4 Design equations and design calculations

The following are design equations used to calculate the resistor values for the voltage Regulator:

$$Vout = Vref\left(1 + \frac{R2}{R1}\right)$$

$$R1 = 270$$

$$3.4 = 1.25\left(1 + \frac{R2}{270}\right)$$

$$R2 = 470$$

2.2.1.3.5 Final design schematics

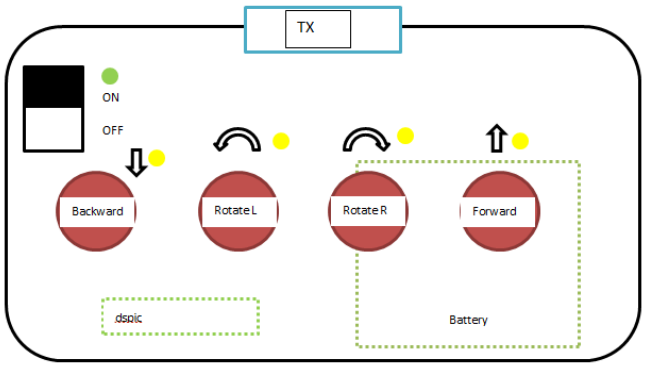


Figure 45: Sensor rotation platform

2.2.2 Hardware implementation

2.2.2.1 Mobile robot

The mobile robot is made up of the sensor system, movement system and processing system. How they interact is described in the Software design (2.2.3).

The mobile robot was built from the bottom up where the motors and wheel placement was designed and built first. Figure 46 shows the servo motors and wheel placement.

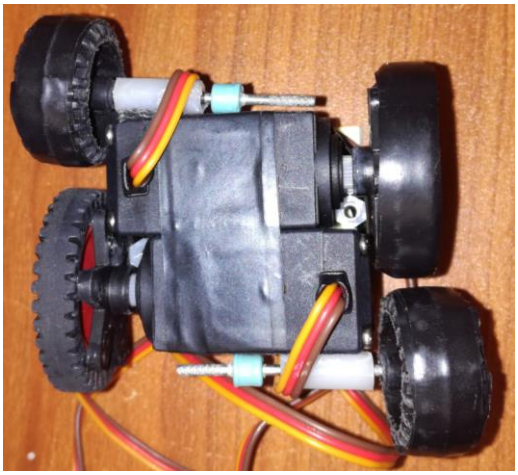


Figure 46: Sensor rotation platform

This design was tested and rejected as discussed in (2.2.1.1.2) .The mobile robot movement system was replaced with stepper motors, two wheels and an omnidirectional wheel. Figure 47 shows stepper motors and wheel placement.

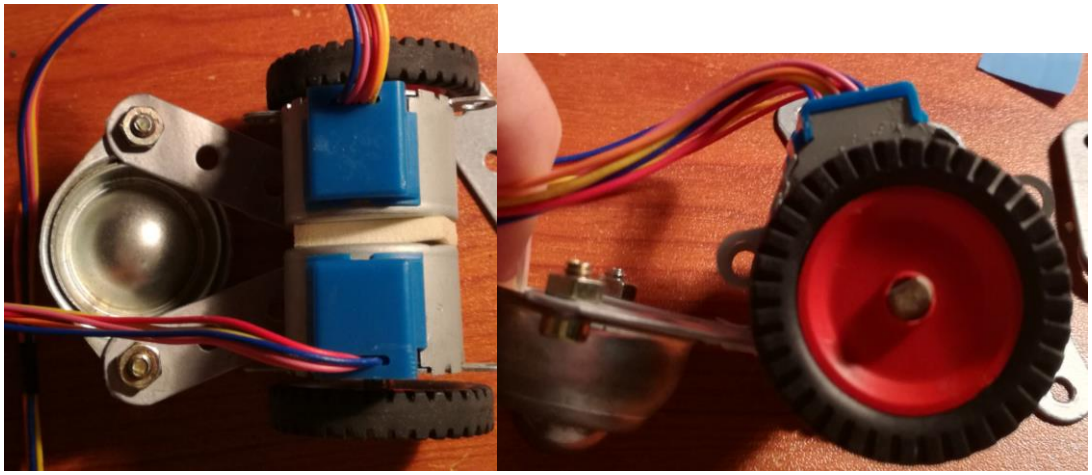


Figure 47: Stepper motor and wheel placement

The sensor system was designed and built using four ultrasonic sensors on a rotating platform. Figure 48 shows the setup of the ultrasonic sensors, with the stepper motor and gear system.

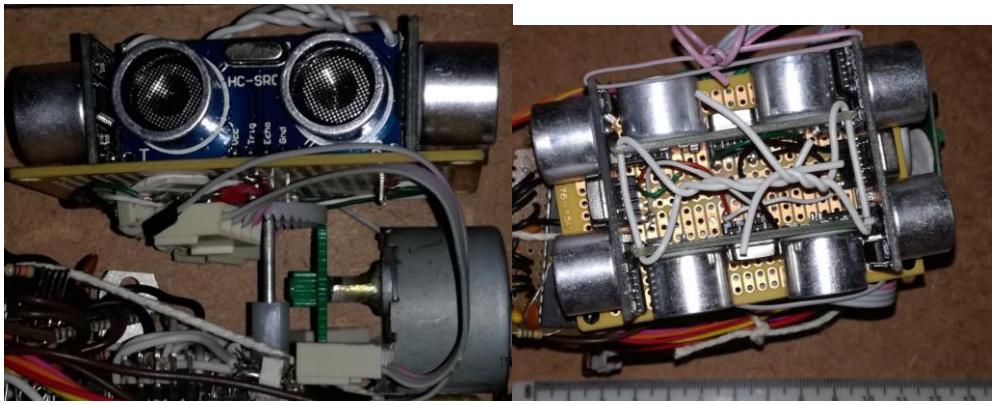


Figure 48: Sensor rotation platform

The processing unit was built on vero board in order to be changed easily. The processing system consisted of two DSPICs, amplifiers, receiver, voltage regulators, and stepper motor drivers ULN2003. The processing system was separated into three smaller vero boards that were placed over each other to keep the size of the mobile robot compact. Figure 49 flow diagram shows how the hardware systems were connected

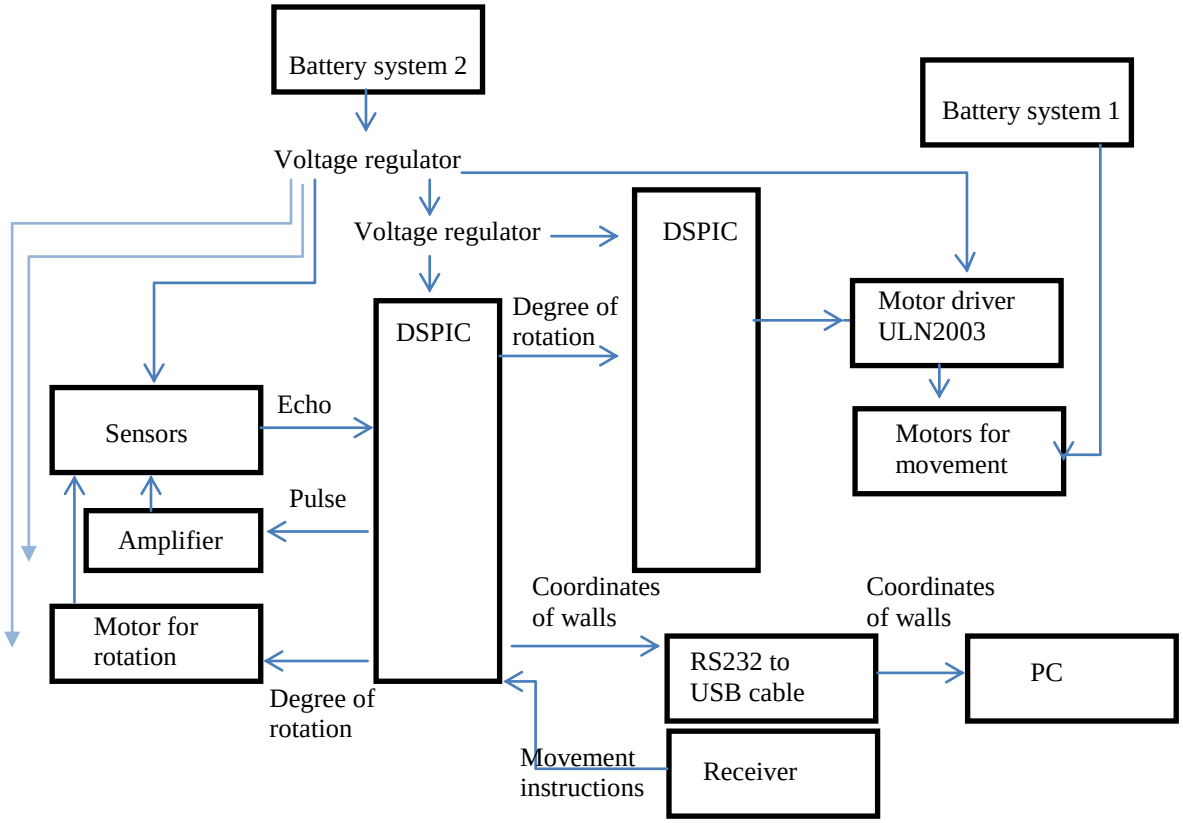


Figure 49: Sensor rotation platform

2.2.2.2 Remote control

The remote control is designed and built with four buttons, one switch, five LEDs, the transmitter, DSPC and voltage regulator as shown in Figure 50 and Figure 51.

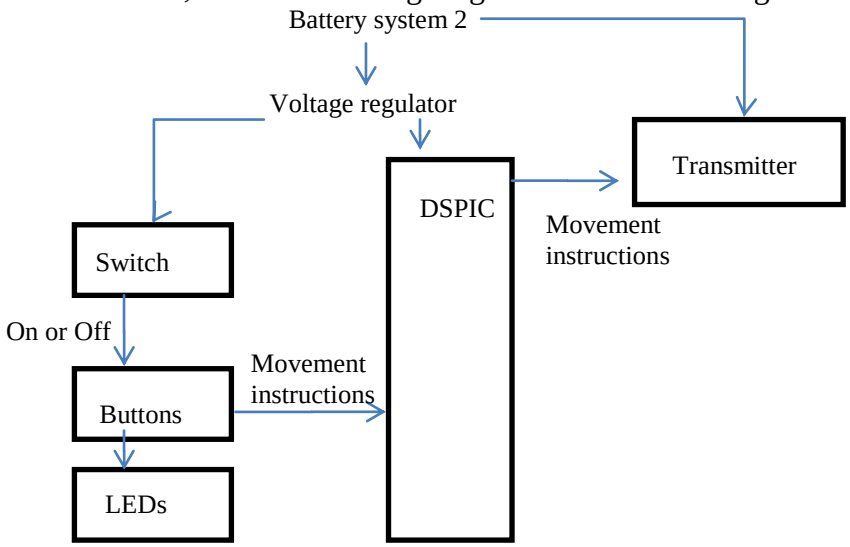


Figure 50: Remote control flow diagram

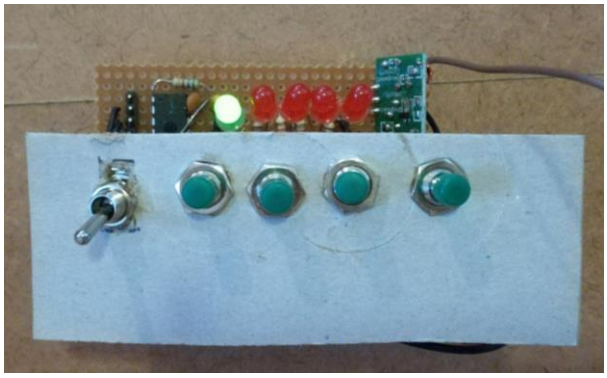


Figure 51: Remote control board

2.2.2.3 Maze

Figure 52 shows possible maze configurations

Figure 52: Maze configuations

2.2.3 Software design

2.2.3.1 Movement

2.2.3.1.1 Design alternatives

If the sensors system had five stationary sensors as mentioned in the mobile robot design alternatives(2.2.1.1.2), then the movement algorithm would work using the information from the sensors to make a decision as to the setup of the walls. As a set number of walls were used, the algorithms would have needed to determine which set of walls was placed in front of the mobile robot. From this decision the mobile robot would move according to how the walls are placed around itself. This limits the algorithm to this specific maze and specific wall combinations. The algorithm would take distances for each of the five sensors and uses these distances against a database to compare which walls are placed around the mobile robot. From this conclusion, the mobile robot would move accordingly around the walls.

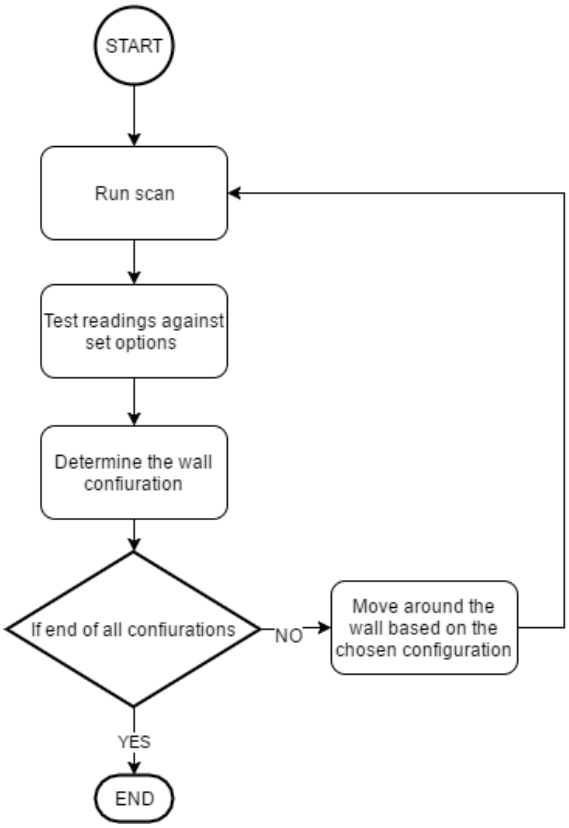


Figure 53: Flow diagrm of rejected sensor sytem

Errors were created from the incorrect decisions being made from the sensors information. This algorithm limited the mobile robot to one specific maze and would not allow for it to be integrated into a different maze or situations therefore the design was rejected.

2.2.3.1.2 Design procedure followed

The movement of the mobile robot was separated into two algorithms. The first algorithm determined how the mobile robot moves around the complex walls directly around the mobile robot and the second determined how the mobile robot moves around the whole maze.

In the first algorithm (the movement of the mobile robot) was dependent on the information from the sensors. The sensors system took readings every 7.5 degrees around 360°. Therefore, 48 distances were saved regarding the information of the walls around the mobile robot. Only specific distances were used to make a decision for the movement of the mobile robot, but all distances were used for the display.

The algorithm to find the distance from the ultrasonic sensors used a counter system where the length of the echo was counted and converted into a millimetre distance. The distances calculated were normalised to the centre point of the mobile robot so that they could be converted into coordinates for display. The algorithm to control the sensors was given as shown in Figure 40. The algorithm sends a short square pulse longer than 10us. The system then starts a counter. If an echo was detected, the length that the echo is that was detected, was used to calculate the distance of the walls. The

algorithm was run five times to average the distances to increase the accuracy of the readings.

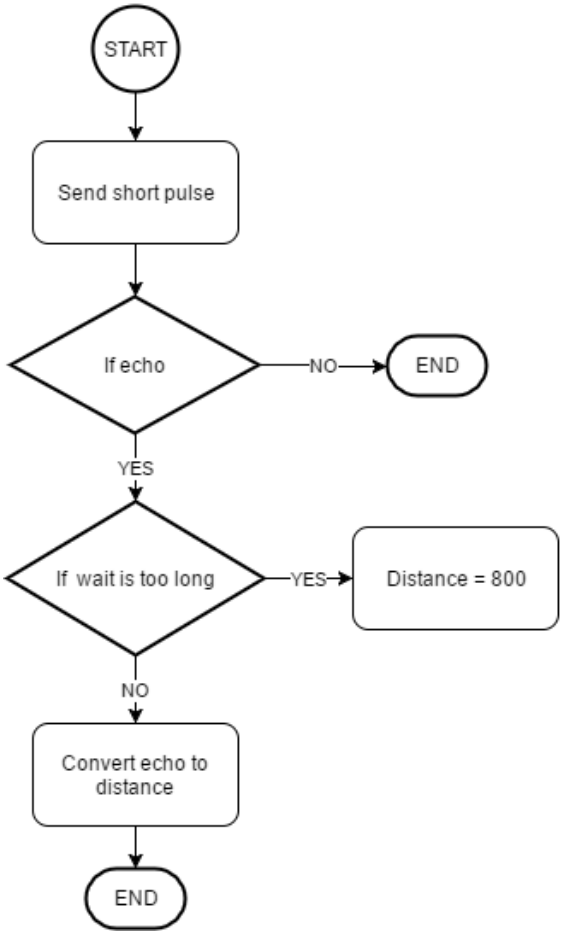


Figure 54: Flow diagram of the working of the sensor

Appendix 1 shows the code used to control the sensors
The algorithm to rotate and save information of the walls is given below:

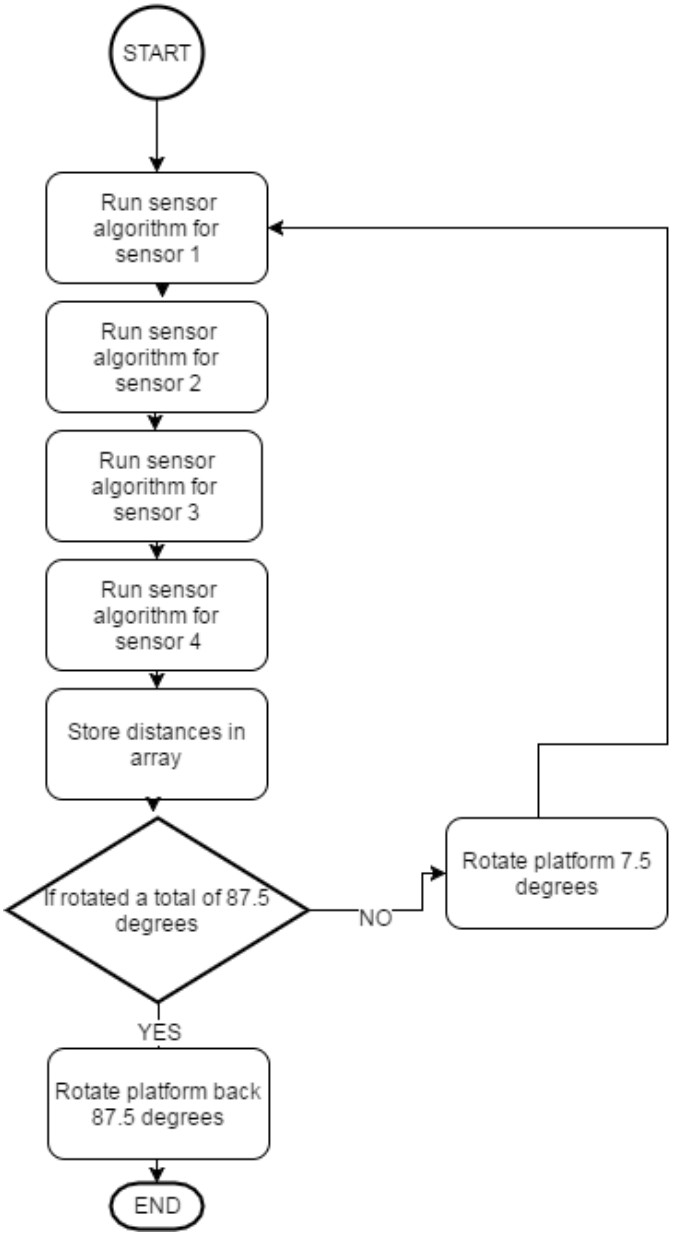


Figure 56: Flow diagram of the sensor system with the rotational platform

Appendix 2 shows the code used to save all of the recorded distances around the mobile robot.

The mobile robot was designed to move around walls, therefore the mobile robot made a judgment as to whether there was a wall to the left or right, or a full wall in front of it; regardless of whether they were 45°or parallel. The mobile robot default travel mode is forward around the walls and therefore the front sensor distances were used to determine what walls were in front of the mobile robot. The mobile robot had three basic x positions in each block of the maze. These were

x = negative, x = 0, or x = positive,

where the negative and positive positions were used to move around the complex walls and the zero position was used to run forward when there were no walls in front.

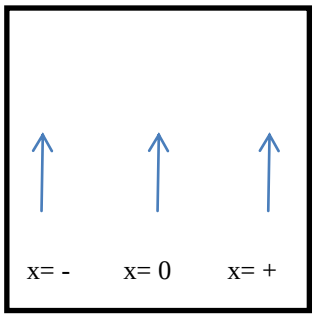
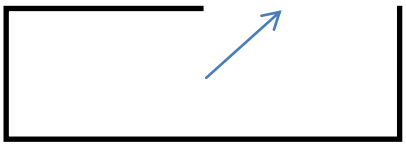


Figure 57: Positions avalible for the mobile robot

Each position had a different algorithm to interpret the data from the sensors and to make decision based on this data. Each position allowed for three movements: to move to the negative, positive, or zero position. For example if x = 0 and a wall was detected on the left, the mobile robot would move right and into position x = + and note this movement.



x= 0

Figure 58: Example of the movement of the mobile robot

The mobile robot therefore moved twice in each block, from the middle to the side or from the side to the middle. Each time the mobile robot moved, it reran a scan to increase the accuracy of the movements and display. The movement of the mobile robot around the complex walls needed to be accurate to decrease errors and decrease error propagation. To control and correct these errors, code was included so that the movement of the mobile robot depended on the distance of the wall in front of the mobile robot. If the mobile robot travelled further from the wall than expected, the mobile robot would travel further to move around the wall and the opposite principle applied when the mobile robot was closer to the wall the expected. The rotation, however, was not corrected and the mobile robot needed to rotate and rotate back by the same degree.

The mobile robot was chosen to travel mainly forward or backwards compared to the reference directions to simplify the algorithms for display and movement. Under specific maze configurations, however, this may not be the most ideal movement and it would therefore take longer to run through. Figure 59 shows how the mobile robot will move relative to its original orientation:

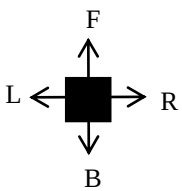


Figure 59: Normalization used for the mobile robot

The mobile robot therefore turned to move around the walls and turned back to keep facing forward or backward.

The second algorithm used to navigate the mobile robot around the maze had conditions. The mobile robot was placed in the maze so that it was parallel with the outer walls and in an x = 0 position in the middle of a block or on the edge of a block. This initial position was used as the normalisation point. All movement and direction would be relative to this position and orientation.
The mobile robot can be started anywhere in the maze, this createed a need for a movement algorithm that ensures the mobile robot will run through the whole maze and around the complex walls of the maze. . To explain how the algorithms works, Figure 60 shows a simple maze and how a robot could travel through it :

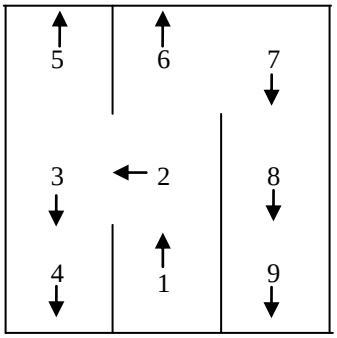


Figure 60: Example of potential robot movement in a maze

The mobile robot starts anywhere in the maze and runs a scan of the walls around it. The mobile robot makes a decision based on the walls around it as to whether it can move left, forward, right or backwards. The mobile robot always moves left first else forward else right else backwards (turns around). When the mobile robot faces the opposite direction the same algorithm is run, however, the first move is left, then backwards, then right.
The mobile robot starts in the Position 1 (P1) and runs a scan. The only option to move is forward, therefore the mobile robot moves forward (P2). The mobile robot runs another scan and finds that it can move left, forward or backward. The backwards movement is cancelled as it is noted that it has been there, and there are no other possible moves to run if it moves backwards. The mobile robot runs left and turns towards (P3). The orientation is now opposite to the original. The mobile robot runs a scan and finds that it can move forward, backwards, and right. Right is noted as a second priority (a route that has not been taken is noted as a first priority). The mobile robot moves towards Position 4. The mobile robot runs a scan and finds that the only movement is backwards. The mobile robot turns around and travels back to Position 3. From previous information stored at Position 3 the mobile robot travels towards Position 5 where a dead end was noted and then travels back to Position 3 where it notes that all routes have been exhausted and it travel back to Position 2. From Position 2, it notes that it can only move forward and moves to Position 6, and by program logic it will travel accordingly from position 6 to position 9. Using this concept the mobile robot will run through the whole maze in a systematic way, noting where it has been.

The algorithm then ran as follows: the scan saves 24 distances from around the mobile robot. From these distances and x positions of the mobile robot, negative, 0 or positive, (where the walls are) is determined and therefore the possible routes available to the mobile robot. The algorithm uses the distances 15° either side of the centre reading: if the distance is less than a set value (maximum wall distance), the algorithm notes this as a wall. If both distances are less than the set wall value, a dead end is noted. The mobile robot will move accordingly to these decisions. The Figure 61 shows where the readings are taken from relative to the mobile robot.

Figure 61: Sensor angle used to make desisions

Figure 62 shows the algorithm for the movement decisions based on the algorithms above.

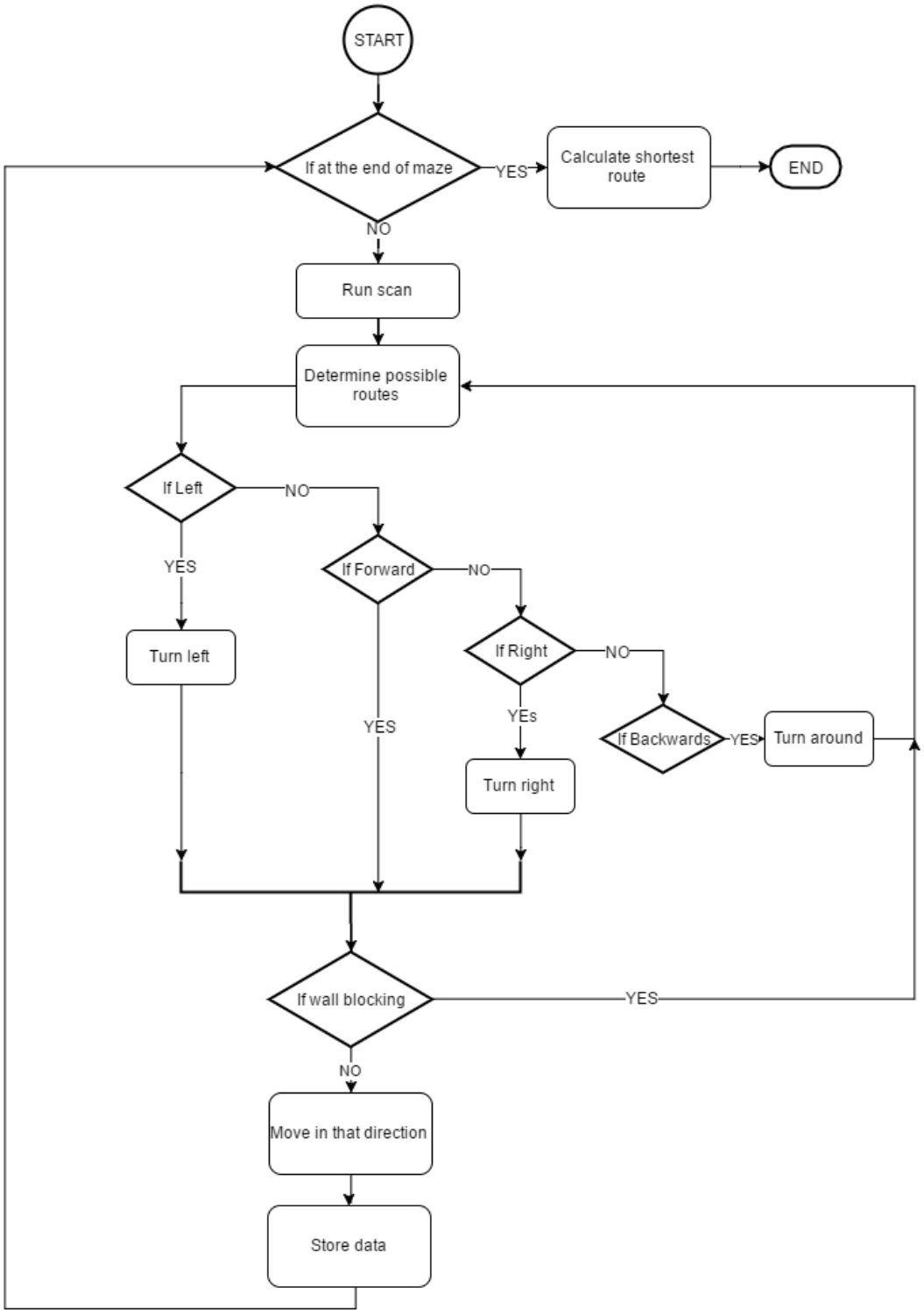


Figure 62: Flow chart for movement algorithm

As the mobile robot runs it keeps track of where it is in the maze relative to the starting position and the mobile robot keeps track of its orientation to normalise the information needed accordingly, such as the display coordinates.

The exits to the maze are noted by the mobile robot running to the exit, running a scan and making a decision based on the sensors' information. This information is used mainly for the shortest route calculation and the mobile robot turns around to continue the route.

2.2.3.1.3 Design calculations

The oscillator frequency was changed for more control over the delays and PWM and UART communication.

Fosc – oscillator frequency
Fcy – device operation frequency
M- multiplication factor
N1-prescale factor
N2-postscale factor
Frc- internal oscillator frequency

$$FOSC = Fin\left(\frac{M}{N1 * N2}\right)$$

$$FOSC = 7.37M\left(\frac{43}{2 * 2}\right)$$

$$FOSC = 80MHz$$

$$Fcy = \frac{FOSC}{2}$$

$$Fcy = 40MHz$$

Appendix 8 shows the calculations used for PWM, when the servo motors were tested.

2.2.3.2 Display

2.2.3.2.1 Design alternatives

If the five sensors system is used as explained in the alternative for the movement (2.2.1.1.2). The combination of the walls around the mobile robot can be compared to the set number of combinations available. Therefore to display the maze, a number from 1 to 40 for each of the 16 blocks in the maze is sent to the PC. Each number represents a specific combination of walls.
This design was rejected as a wrong decision created a large error in the display of the walls and there are a few combinations that would have been difficult to interpret such as the example given below.

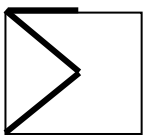


Figure 63: maze wall configuation

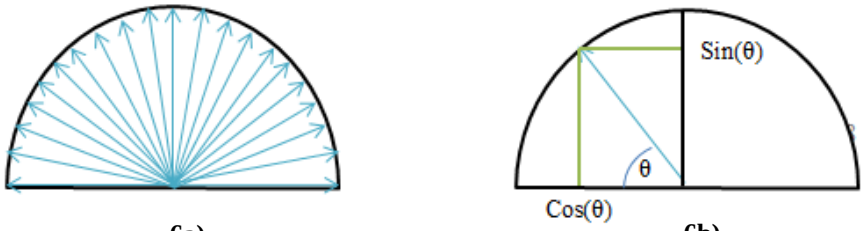
2.2.3.2.2 Design procedure followed

The display algorithm used the distances recorded from the ultrasonic sensors
The distances are converted into a coordinate system using the angle that the distance was taken (every 15°). The following shows the equation used to find the coordinates

$$(x; y) = (distance * \cos\theta\ ;\ distance * \sin\theta)$$

Equation 6 : Distance calculation

The following figure shows how the distances are taken around the mobile robot and the how each of these distances is converted into a coordinate system:



6a) **6b)**

Figure 64: Sensor angles and (x;y) coordinates

If the distance measured was greater than a threshold value the distance would be ignored as it is either a gap in the walls or a false reading from the angle of the wall being too sharp. As the mobile robot moved, the coordinates are normalised to the movement. Where the movement of the mobile robot was recorded as an (x,y) coordinate system. The start is saved as (0,0) and from there, if the mobile robot moves forward the y coordinate will change and if the mobile robot moves left or right, or around walls the x coordinate will change. These coordinates are used to shift the coordinates calculated from the sensor system. The coordinates are also normalized to the display where the display used the following coordinate system and the initial coordinates from the first position are placed around 350 and 350, this allowed the coordinate system to be plotted from this point where the mobile robot ran left, right , forward or backward:

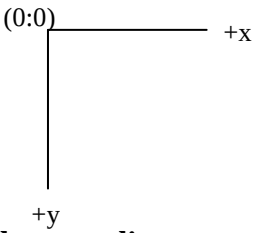


Figure 65: Display coordinate system

Appendix 3 shows how the coordinates are calculated and normalized to the orientation of the mobile robot. This code is integrated into the scan code used for the sensor system.

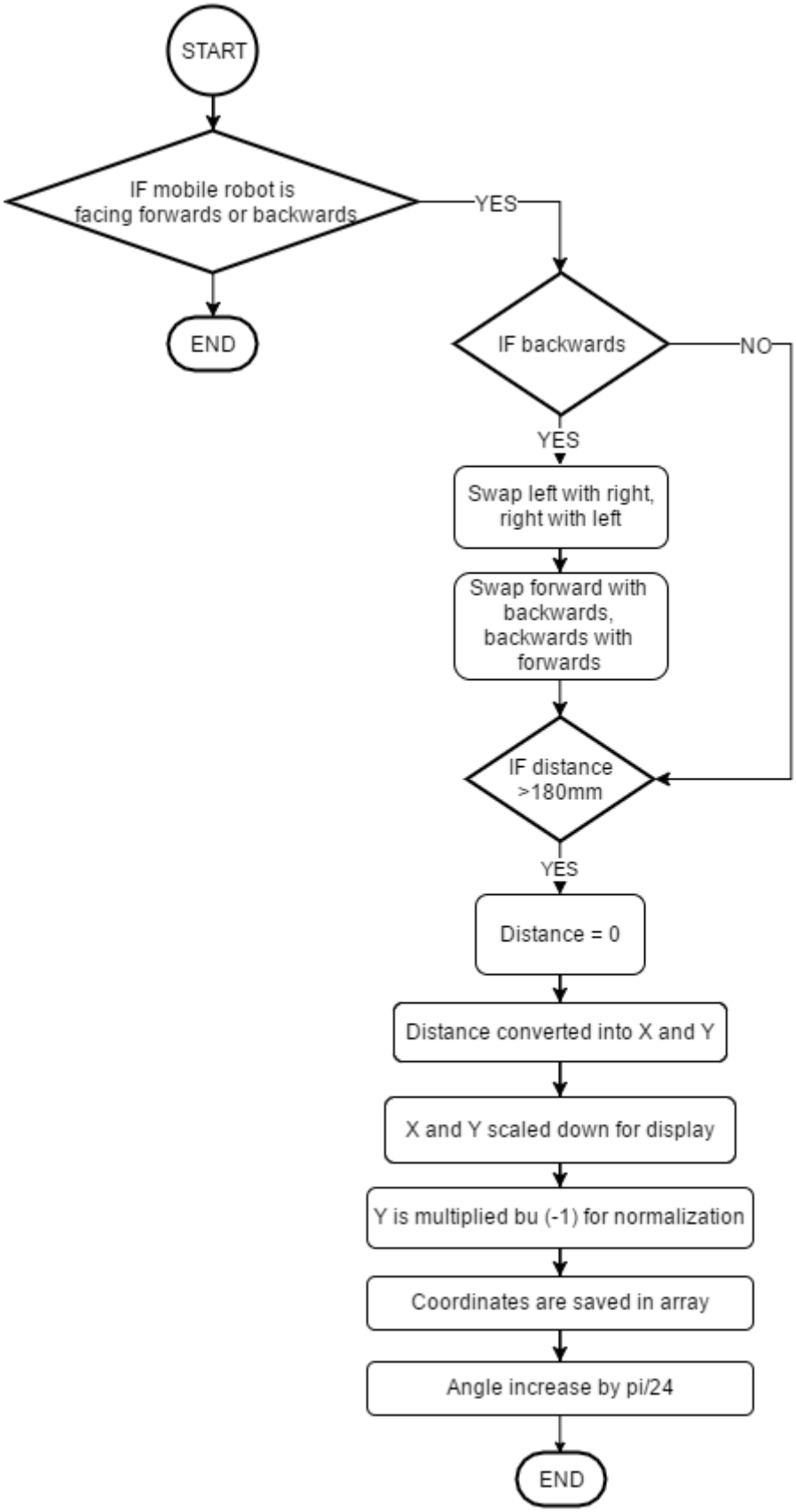


Figure 66: Flow diagram of the normalization algorithm

Appendix 4 shows how the coordinates are normalized to the position of the mobile robot in the maze.

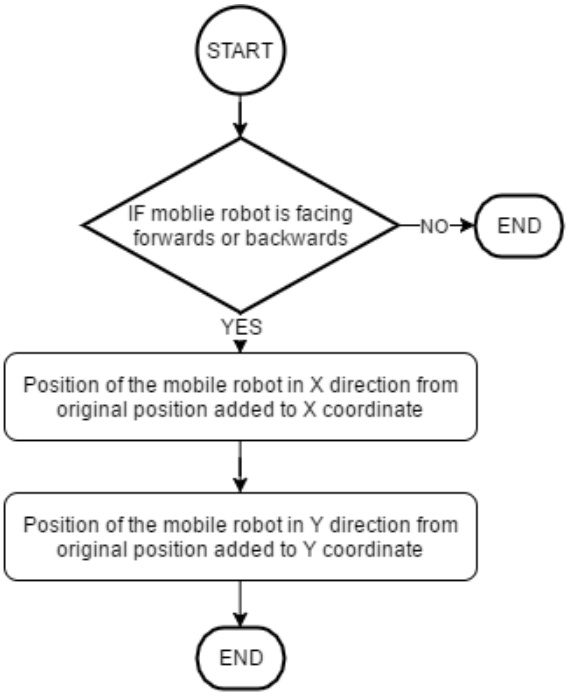


Figure 67: Flow diagram of the normalization for the movement

The number of coordinates taken is 2688 = 4 x 7 x 48 x 2. These coordinates are stored in an array and send via UART to a PC. UART works by sending ASCII characters as shown in the ASCII table.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Figure 68: ASCII table [20][1]

The maximum value that the coordinates can calculate is 999. However, when sending a value greater than 9, such as 10 or 234, two or three asci characters would be sent. For example, 234 is sent, three ASCII characters are sent to create 234.

234 -> 2=50
3=51
4=52

For each coordinate sent, the value was converted into three ASCII characters and sent with '-' to separate the values when received on the PC.

Appendix 5 shows how the UART sends the coordinates:

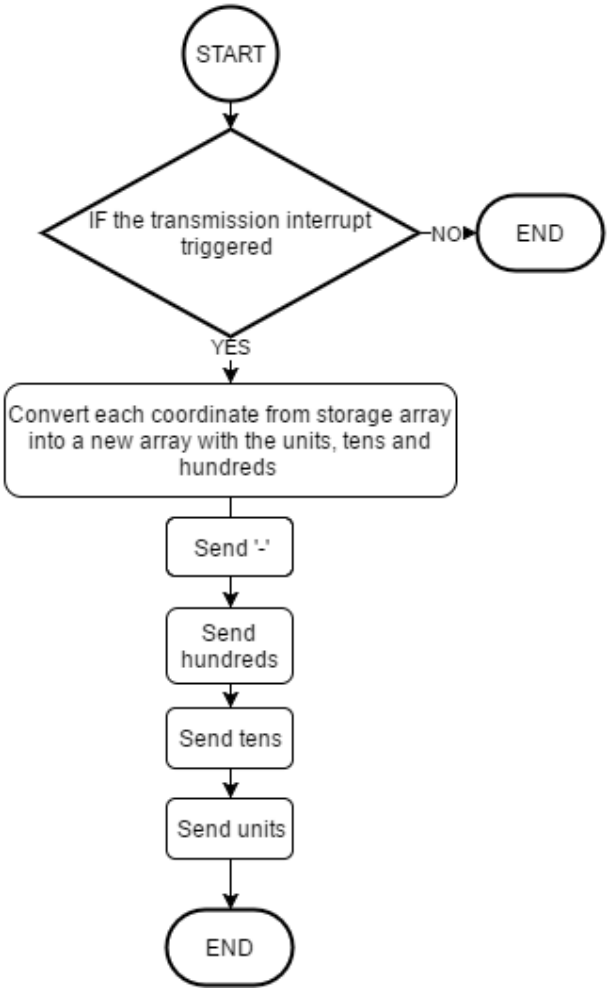


Figure 69: Flow chart of UART communication

The program Putty is set up to receive the data from the RS232 to USB cable and store the data in a text file. The coordinates in the text file are read into an array in Dev-C++ and are then plotted as there x and y pairs as a small circles on a display. The start point of the display is in the middle of the screen and allows for four rows on either side to be plotted. This allows for the mobile robot to start anywhere in the 4x4 maze and plot the whole maze on the display.

2.2.3.2.3 Design calculations

UART baud rate for 9600bits/second
U1BRG = baud rate generator prescaler
fcy=device operation frequency

$$U1BRG = \frac{fcy}{16 * baud\ rate} - 1$$

$$U1BRG = \frac{40M}{16 * 9600} - 1$$

$$U1BRG \approx 260$$

2.2.3.3 Shortest route algorithm

2.2.3.3.2 Design procedure followed

The shortest route algorithm uses a similar algorithm to the movement algorithm where it runs left first, then forward, then right, then backwards.
As the mobile robot moves, it keeps track of where in the maze it is in comparison to its starting position. It also tracks where the mobile robot can move in each block. When the mobile robot finds an exit or entrance to the maze, it notes where the exit is. Using this information the shortest route can be calculated.
The algorithm scans all of the blocks to find an exit or entrance point. It makes this point the starting point. From the information of the possible movements, the algorithm runs left only until a dead end or an exit is encounter. If a dead end is found, the mobile robot runs back to the last point where it could move differently and move in that direction until another dead end or an exit.
If there are two routes to the exit the distances of each route is calculated and compared. The shortest route is chosen. If there are two routes to the exit this means there is a loop in the maze.

The algorithms works as follows:

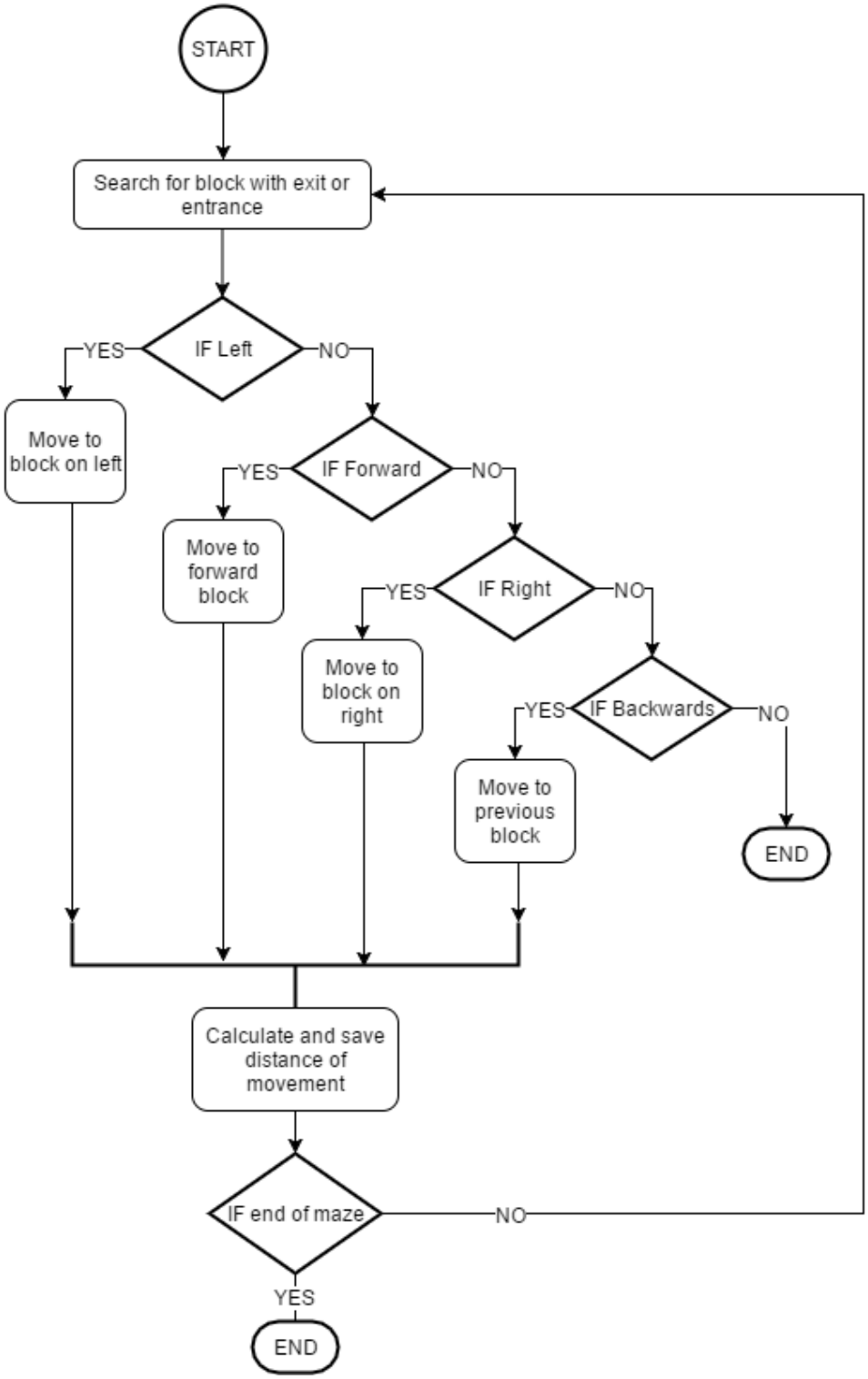


Figure 70: Flow chart of shortest route algorithm

2.2.3.3.3 Design calculations

The distance that the mobile robot has travelled is calculated using the following equation:

$$distance = \sqrt{|x - xprev|^2 + |y - yprev|^2}$$

Equation 7: Disance between two points

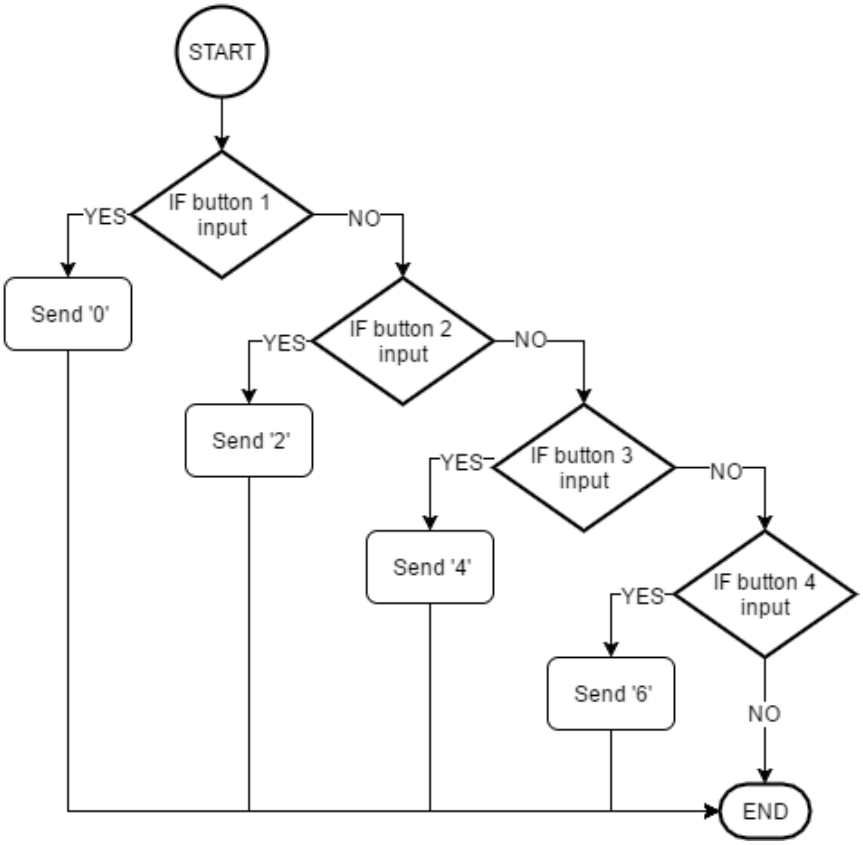
2.2.3.4 Remote control

2.2.3.4.2 Design procedure followed

The software for the remote control uses UART from one DSPIC to another through a wireless communication system. Figure 71 shows the code used to send the information from the buttons on the remote though UART.
The algorithm looks for a high input from the buttons and sends either a 0, 2, 4 or 6 depending on which button is pressed. The following shows how the remote receives and interprets the informaion from the remote:

The algorithm inputs the data received and checks to see if the data sent corresponds to the data expected. The data expected is tested from the wireless communication system. If the input corresponds with the expected command the algorithm sends information to the second DSPIC to control the motors movements.

Transmitter:



Receiver:

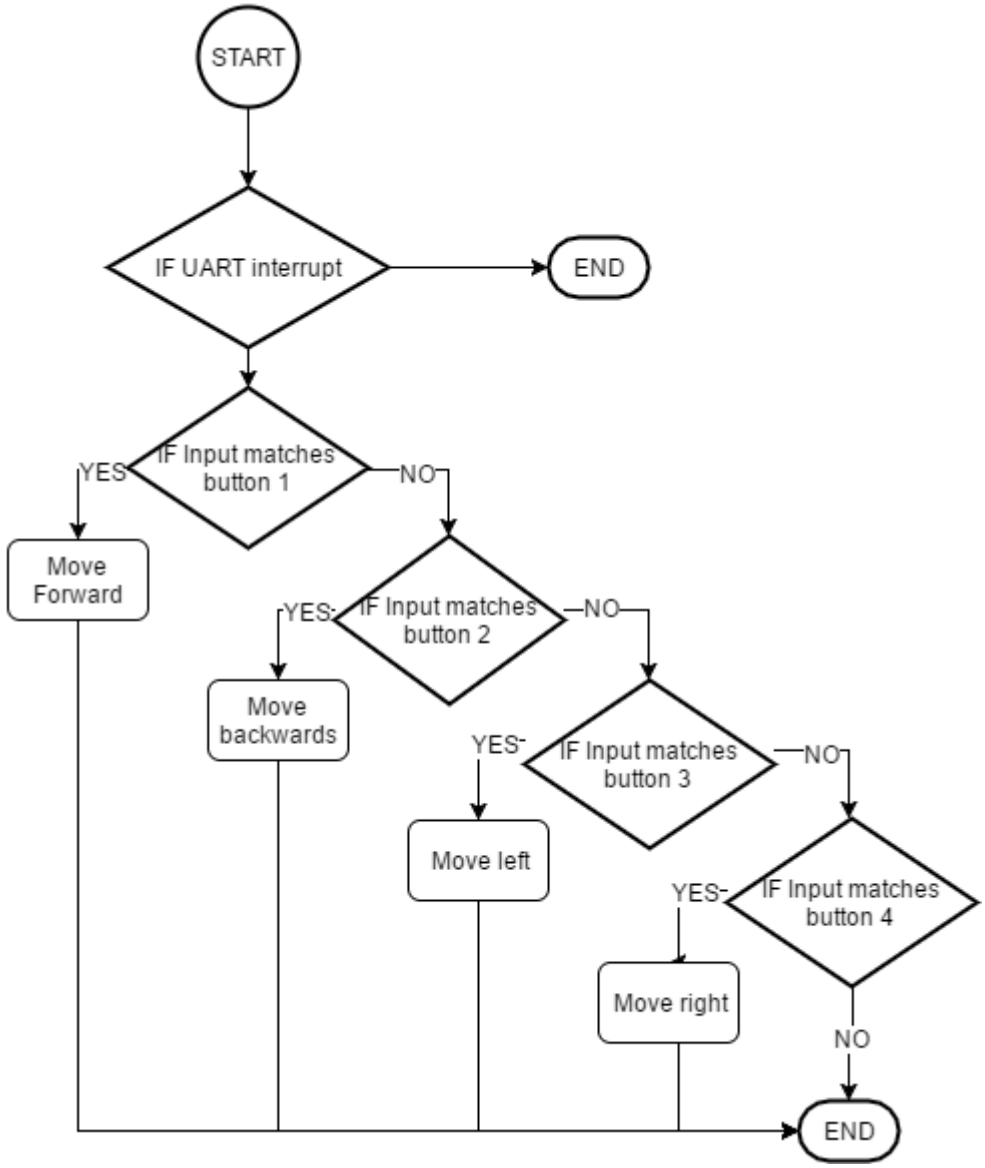


Figure 71: Transmitter and reciever flow diagram

The mobile robot's sensor system was linked to the remote control instructions. The sensor system repositioned so that the front sensor was parallel with the mobile robot. As the mobile robot responded to the inputs from the remote control, the sensors ran to check the distance of the wall around the mobile robot. If the mobile robot was being driven directly into a wall, using the remote control, the mobile robot would stop and wait for a command that would move it away from the wall.

2.2.4 Software implementation

The sensors system, movement system, display system, shortest route system and remote control system need to be integrated into one processing system.

The following algorithm in Figure 72 shows how the different pieces of code are integrated:

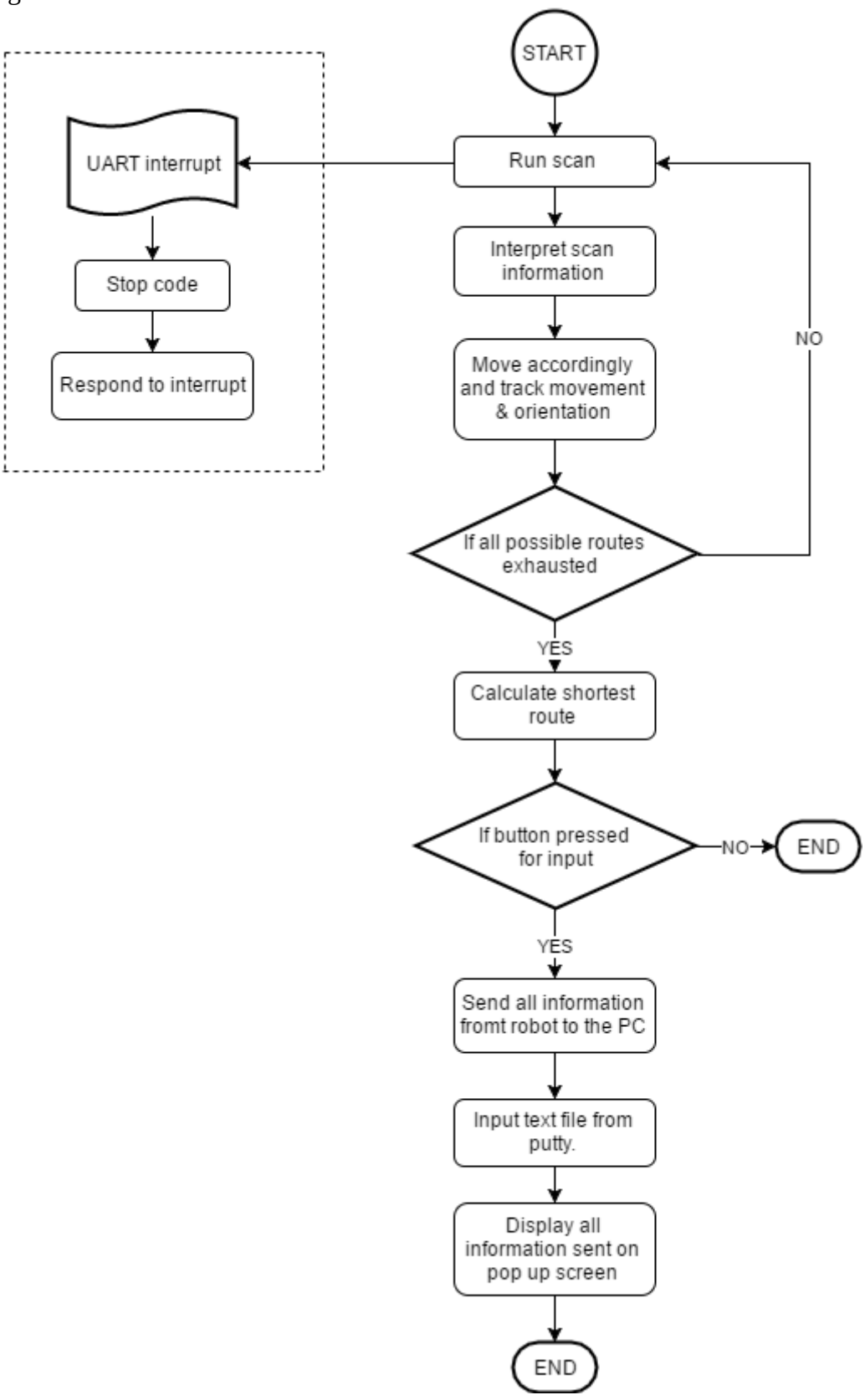


Figure 72: Flow diagram of the whole software system

Run scan- runs the sensors system where the sensors take distance readings ever 7.5 degrees. The motors controls the movement of the rotating platform and therefore moves and stops every 7.5 degrees. The sensors take two distances recordings every stop and average these readings to increase the accuracy

Interpret scan information- this algorithms decides where the walls of the maze are relative to the mobile robot and how to move around the walls.

Move accordingly and track movement and orientation – this algorithms used the information of where the walls are to move either left, forward, right or backwards around the walls. The movement and orientation of the mobile robot is tracked and stored.

When all possible routes exhausted; calculate shortest route – this algorithm checks to see if the mobile robot has run though all the possible routes available to the maze. If the mobile robot has exhausted its routes the mobile robot moves on to using the information stored to calculate the shortest route. The shortest route algorithm is run to find and store the shortest route through the complex maze.

When input -send information for display of walls and shortest route to PC- the mobile robot waits for an input to note that the UART serial communication cable has been connected. The algorithm then runs the UART to send all of the information of the walls and shortest route to the PC into a text file from Putty.

Input text file from Putty. Display all information sent on pop up screen- this algorithm is run in Dev-C++. The information from the UART communication in the text file in input into an array to be displayed. The information in the array is displayed using the graphics.h library by displaying circles in a pop up screen. This created the 2D display of the maze and shortest route.

If UART interrupt: stop code and respond to input- if the remote is turned on and used to send information to the mobile robot the input causes an interrupt where the information from the remote is interpreted to control the movement of the mobile robot. The sensors are reposition to be 90° to the wheels and the sensor are run between responding to the remote inputs. When the distance mobile robot is 4cm from a wall the mobile robot will not respond to instructions that will allow the mobile robot to drive into the wall.

2.2.5 Final system integration and testing

The software system and hardware system are integrated. The DSPIC is the main link between the two systems and the majority of the software is run from the DSPIC. The DSPIC links the hardware through code.

2.3 DESIGN SUMMARY

This section summarises the project tasks and how they were implemented (see table I).

Task	Implementation	Task completion
Design mobile robot	Sensors, motors and processing units where researched. The layout combining the three systems was designed to be compact with three levels that allowed the sensor system to rotate 90°.	Complete
Build mobile robot	The motors were connected to wheels and the processing unit. The processing unit was placed on its own level above the motors. The sensors were placed at 90° on a platform. The platform was kept above the processing unit using a pole. A stepper motor was placed below the platform, on the processing units level and connected to the platform through a 90° gear system. The battery system was place within the three levels	Complete
Design maze	The maze was designed to allow for different wall configurations to be placed on one base. A prototype was designed and built first to test the theoretical size calculated. The walls options are full length walls, half walls and walls that can be placed at 45°.	Complete
Build maze	The maze was built after the prototype and mobile robot had been completed and tested. The base was split into two. The base was reinforced. Holes were made in the base to allow for walls to be placed in the maze connected to a clip. The clips were screwed into the base where walls were desired.	Complete
Create movement algorithm	The algorithm started by being able to interpret the sensor readings of the walls in front of the mobile robot and move the mobile robot around the walls in front of itself. The algorithm was then incorporated with the movement algorithm that would allow the mobile robot to run through the whole maze from any position. This algorithm used the principle of the depth first algorithm to run as far left as possible then track back and run a different route as far left as possible.	Complete – not fully tested
Create display algorithm	The display algorithm interpreted the information of the walls directly around the mobile robot into coordinates and then was incorporated	Complete

	with the movement of the mobile robot through the maze to shift and normalize the coordinates to the movement and orientation of the mobile robot.	
Create shortest route algorithm	The shortest route algorithm was created after the movement, of the mobile robot had been completed. the shortest route algorithm uses the depth first algorithm as a base and incorporated the saved data of the movement and therefore the walls of the maze. From the movement data the distance are be tracked when the algorithms is run	Complete – not fully tested
Design remote control	The remote control was designed to allow for four push buttons and one switch. The wireless communication device was researched.	Complete
Build remote control	The remote was built using the same DSPIC. Push buttons and the switch were liked to LEDs for display. The DSPIC was incorporated with the buttons, the switch and the transmitter. A simple antenna was built using a wire and connected to the transmitter. The batter system was added between the buttons and the DSPIC.	Complete

TABLE 1

3. Results

3.1 SUMMARY OF RESULTS ACHIEVED

Description of requirement or specification (intended outcome)	Actual outcome	Location in report
Mission requirements of the product		
The shortest route must be automatically calculated and demonstrated within 4 minutes after the mobile robot has mapped the maze.		Section 1.4.1
All the walls must be plotted on the PC with an error of less than15%.	The walls plotted have an error of less than 15% for simple wall configuration with no sharp corners.	Section 1.4.1
The mobile robot needs to respond correctly to instructions from the remote control at a maximum time of 1 second.	The mobile robot responds correctly to each of the four instructions. The time that the mobile robot takes to respond is less than 1 second.	Section 1.4.1
The mobile robot must move independently around the walls with a minimum distance of 3 cm from the walls.	The minimum distance that the mobile robot moves around the walls is 2cm	Section 1.4.1
The maze must be at least 10 times larger than the mobile robot.	The size of the mobile robot is 6cm x 10.5cm and the size of the maze is 112cm x 112cm therefore the maze is approximately 14.66 times bigger than the mobile robot.	Section 1.4.1
Field conditions		
The product should work correctly in a temperature of 5 to 40°C.	All of the products used in the mobile robot and remote have temperate rating between 5 and 40°C. Therefore the product works in the temperature range.	Section 1.4.2
The product will work with indoor lighting conditions (400 to 600 lux).		Section 1.4.2
The product will work with ideal outdoor lighting conditions (10 000 to 100 000 lux).		Section 1.4.2
The product will work with wind levels of 0 to 7 km/h.		Section 1.4.2
The product will work with indoor and outdoor background noise conditions of up to 60 dB.		Section 1.4.2

Specifications		
FU1. The battery must power the mobile robot though mapping the entire maze from the start to the end twice.		Section 1.4.3
FU1. The mobile robot dimensions must be less than 20 cm x 20 cm.	The mobile robot dimensions are 10.5cm x 6cm which is less then 20cm x 20cm.	Section 1.4.3
FU1.1. The mobile robot must run though entire maze in under 6 minutes.		Section 1.4.3
FU1.2. The remote, must be wireless and work within a radius of 10 m.	The remote is runs with a wireless communication system and a battery pack. The remote control has a radius of 30cm.	Section 1.4.3
FU1.3. The sensors must detect the distance of a wall with an error of less than of 14%.	The sensors detect the distance of the walls at angles less then 30° with an error of less than 14%.	Section 1.4.3
FU1.4. The mobile robot, must drive at a speed of greater than 5 cm/s and slow down to stop.	The mobile robot runs at a maximum speed of 4cm/s.	Section 1.4.3
FU1.4. The mobile robot must turn at a speed of at least 45° in 10 seconds.		Section 1.4.3
FU1.5. The storage needs to be at least 10 Mb in order to store all the relevant data.	The storage used is incorporated in the DSPIC. The storage available is 16Mb on the DSPIC.	Section 1.4.3
FU1.6. The mobile robot must send the stored data to a PC though a cable within 30 seconds.	The mobile robot sends the stored data to the PC in with in 30seconds	Section 1.4.3
FU2. The remote must be battery driven for at least to 20 minutes.	The remote can be battery driven for at least 20 minutes	Section 1.4.3
FU2. The remote must have five movement options: forward, backward, rotate left, rotate right and stop.	The remote has four movement options. With the stop option being controlled with the push button system and the on off switch	Section 1.4.3
FU3. The display must plot all the walls with an error of less than 15%.	The walls plotted have an error of less than 15% for simple wall configuration with no sharp corners.	Section 1.4.3
The height of the walls of the Maze must be less than 15 cm, and the walls must be thinner than 1 cm.	The height of the walls in the maze is 14cm The width of the walls is 0.3mm	Section 1.4.3

Deliverables		
Distance sensors	The student bought the sensors	Section 1.5.1
Driving system	The student designed the driving system and bought the motors and wheels	Section 1.5.1
DSP	The student bought a DSPIC instead of a DSP	Section 1.5.1
Batteries	The student bough the batteries	Section 1.5.1
Algorithm to process remote control input data	The student designed and implemented the algorithm to process remote control input data	Section 1.5.1
Algorithm to process sensor input data	The student designed and implemented the algorithm to process sensor input data	Section 1.5.1
Algorithm to calculate the shortest route	The student designed and implemented the to calculate the shortest route. Using the left only principle as a guide.	Section 1.5.1
Design of mobile robot to hold sensors, driving system, battery system and DSP	The students designed and implemented the mobile robot robot to hold sensors, driving system, battery system and DSPIC.	Section 1.5.1
Design of the maze	The student designed and implemented the maze	Section 1.5.1
Remote control	The student designed and implemented the remote control	Section 1.5.1
Wireless communication system	The student bought and used the wireless communication system	Section 1.5.1
Storage device	The student did not need a storage device as the DSPIC had enough storage for the information needed.	Section 1.5.1
Cable for sending data to PC	The student bought an RS232 to USB serial communication cable.	Section 1.5.1
PC software	The student used Putty and Dev-C++ as the PC software to display the maze	Section 1.5.1

TABLE 2

3.2.1 Experiment 1: Qualification test 1 of shortest route algorithm

3.2.2 Experiment 2: Qualification test 2 of the display accuracy

3.2.2.1 Qualification tests
The objective of this test is to find the accuracy and therefore the error of the display system used in this project to display the maze walls. All the walls must be plotted on the PC with an error of less than15%.

The equipment used :
-a laptop with Mplab, Dev-C++ and Putty software.
-the mobile robot built for the project

Test procedure:
-the maze walls were set up to display different wall configurations between two blocks of the maze.
-the mobile robot was placed in the middle of the first block.
- the algorithm for the movement of the mobile robot was used where after three movements the mobile robot sent the information from the sensors over the three movements via an RS232 to USB cable.
-this information was stored in a text file to be read by the Dev-C++ algorithm.
-the Dev-C++ algorithm was ran and the output on the screen is saved.
Calculate error

The number of distances taken at each angle is 5 where these distances are averaged to increase the accuracy of the readings

3.2.2.2 Results and observations

The maze was set up as follows: a back wall is added

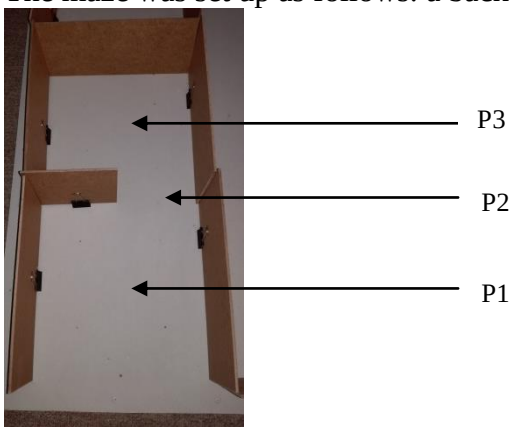


Figure 73: Maze configuration with half wall

The mobile robot moves automatically from position one to position two and finally to position three. Each time the mobile robot runs a scan of the walls around itself, it saves this information.

Figure 74 was the output displayed on the screen of the maze configuration in Figure 73

Figure 74: Display configuration with half wall

The basic layout of the walls can be seen with erros mainly in the corners. This is expected as ultrasonic sensors work best with a flat surface and the echo from the detail in the corners is lost or returns with errors.

The display is set up so that the mobile robot can start anywhere in the maze and the full maze will be displayed on the screen.

The error is calculated to be: approximately 14%

The maze is changed to the following set up:

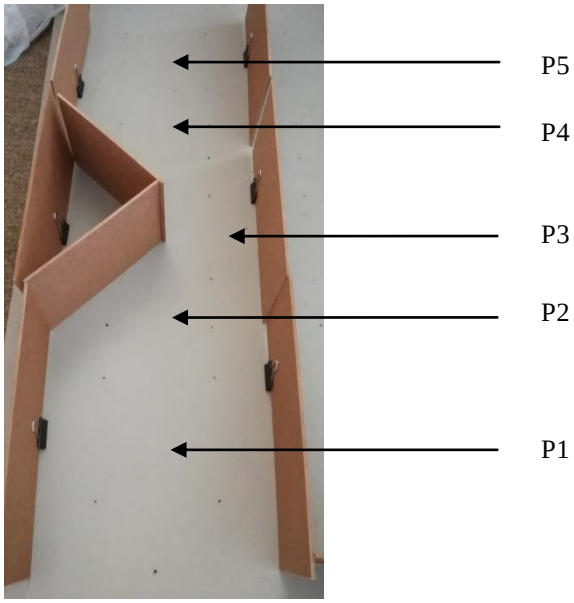


Figure 75: Maze configuration with 45 degree wall

The mobile robot moves from position one to position five, and each time the mobile robot moved it ran a scan of the detail of the walls around it.

Figure 76 was the output displayed on the screen of the maze configuration in Figure 75.

Figure 76: Display configuration with 45 degree wall

The following two experiments showed the limitations of the senor system and how large errors can occur with the display

The maze was set up as follows:

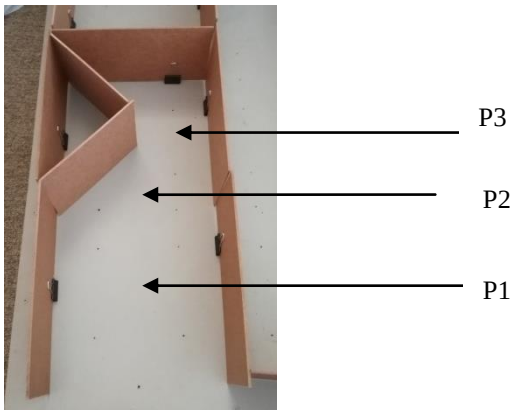


Figure 77: Maze configuration with 45 degree wall and dead end

Figure 78 was the output displayed on the screen of the maze configuration in Figure 77.

Figure 78: Display configuration with 45 degree wall and dead end

A large amount of detail of the maze walls was lost, when the mobile ran a scan at position 3. The mobile robots sensor system cannot move into this space as the angles of the walls are too sharp relative to the ultrasonic sensors from its position at 3.
If more than 16% of the walls are plotted in the display the error of the display will be greater than 15%.

The maze was changed to the following set up:

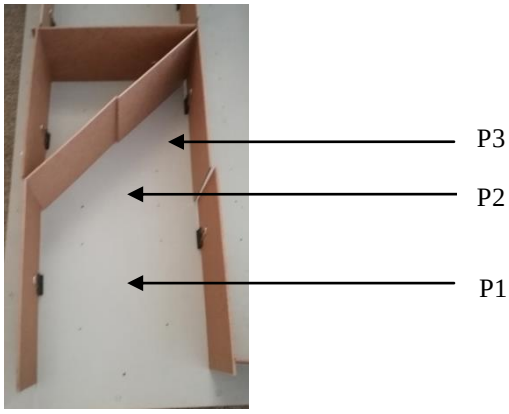


Figure 79: Maze configuration with closed 45 degree wall

Figure 80 was the output displayed on the screen of the maze configuration in Figure 77.

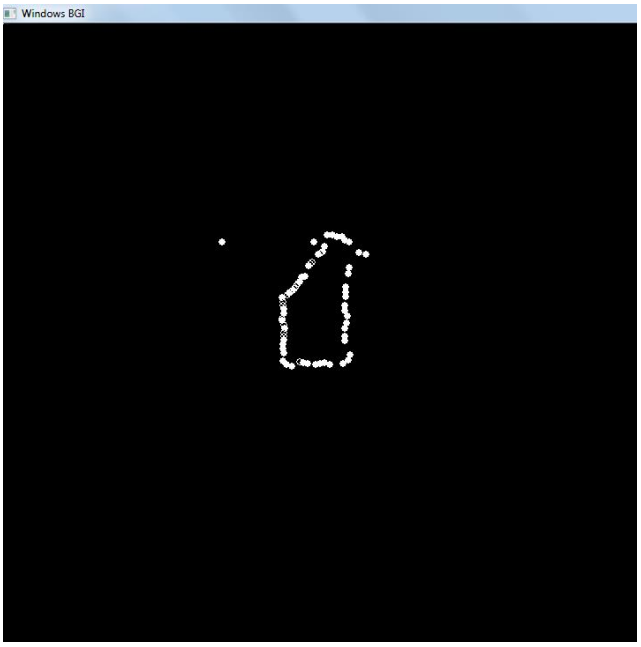


Figure 80: Display configuration with closed 45 degree wall

The detail of the sharp corner was lost in the display for the same reasons as mentioned above.
If more than 20% of the walls are displayed incorrectly the display will have an error greater than 15%.
The more complex the maze was made, the less accurate the readings from the sensors system especially with walls that have a sharp angle.

The errors are calculated using the following equation.

$$\text{Error} = \text{Experimental value} - \text{"true" or theoretical value}$$

$$\text{Percent Error} = \frac{\text{Error}}{\text{Theoretical value}} * 100$$

Equation 8: Percentage error[21]

3.2.3 Experiment 3: Qualification test 3 the remote response

3.2.3.1 Qualification tests

The objective of this test was measure the response time of the mobile robot to an instruction from the remote control and to test if the mobile robot responds correctly to each of the remotes inputs. The mobile robot needs to respond correctly to instructions from the remote control in a maximum time period of 1 second.

Equipment:
- the mobile robot built for this project was used to test the remote.
- a stop watch

Experimental procedure:
The switch on the remote was switched to on.
The forward button of the remote was pressed
The time for the mobile robot to respond was timed
The mobile robots movements was observed

Expected results:
The remote control uses UART communication with a baud rate of 9600bits/sec. Eights bits represent different ASCII characters and are sent with a stop bit via UART communication.

9600bit/sec = 104.16us/bit

For 9 bits:
=937.5us

The operating frequency of the DSPIC is set to 40MHz, therefore the instruction cycle is 2.309ns.

The total time to send and receive, and interpret an instruction was therefore less than 1 second.

3.2.3.2 Results and observations

The time for the mobile robot to respond to the commands of the remote was:
The mobile robot moved correctly to each of the four commands and stopped when the push buttons were released.

3.2.4 Experiment 4: Qualification test 4 the distance from the walls

3.2.4.1 Qualification tests

The objective of this test was to find the minimum distance the mobile robot can move around the wall configurations. The mobile robot must move independently around the walls with a minimum distance of 3 cm from the walls.

3.2.4.2 Results and observations

The width of the mobile robot was measured at 6cm. Therefore, for a half wall configuration, the mobile robot had 4cm either side to move around this wall configuration. Figure 80 shows that the mobile robot has 4 cm either side when moving around this wall configuration.

Figure 80: Distance of mobile robot around the maze walls

The wall configuration in Figure 81 wad the configuration used to calculate the size of the maze as the mobile robot needed to be able to move around this configuration.

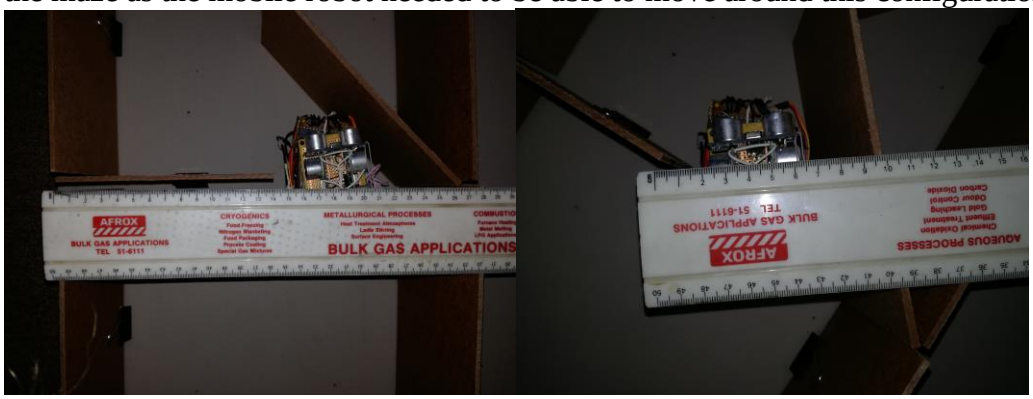


Figure 81: Distance of mobile robot around the maze walls

Figure 81 shows that the mobile robot does not move around this wall configuration with the minimum distance of 3cm.

3cm was made a specification as it allowed for the senor system to not create errors for the display or movement. Most ultrasonic sensors have a minimum range from

3cm. The ultrasonic sensors used in this project however have a minimum distance of 2cm and therefor will not create errors in the display or movement.

3.2.5 Experiment 5: Qualification test 5 the size of the maze

3.2.5.1 Qualification tests
The objective of this test is to verify that the size of the maze is at least ten times larger than the mobile robot. The maze must be at least 10 times larger than the mobile robot.

Experimental procedure:
The dimensions of the mobile robot are taken
The dimensions of the maze are taken
The dimensions are compared

3.2.5.2 Results and observations

Figure 82 shows the size of one of the blocks in the maze to be 28cm x 28cm, and the outline of the whole maze relative to one block.
Therefore the area of the maze is calculated to
112cm x112cm
=12544cm^2

Figure 82: Size of the maze

Figure 83 shows the length of the mobile robot at 10.5cm and the width to be 6cm.
The area of the mobile robot is calculated:
6cmx10.5cm
=63cm^2

Figure 83: Length and width of the mobile robot

- the length of the maze is 18.66 times larger than the mobile robot.
- the width of the maze is 10.66 times larger than the mobile robot.
- the area of the maze is 199.11 times larger than the mobile robot.

The maze is therefore ten times larger than the mobile robot.

Figure 84 shows the size of the mobile robot relative to the maze.

Figure 84: size of mobile robot compared to the size of the maze

Specification results

3.2.6 Experiment 6: Qualification test 6 The battery must power the mobile robot though mapping the entire maze from the start to the end twice.

3.2.7 Experiment 7: Qualification test 7 The mobile robot dimensions must be less than 20 cm x 20 cm.

3.2.7.1 Qualification tests

The objective of this test is to verify that the mobile robot is less than 20cm x 20cm.

Equipment:
-ruler

3.2.7.2 Results and observations

The area of the mobile robot is calculated:
6cmx10.5cm
$=63cm^2$

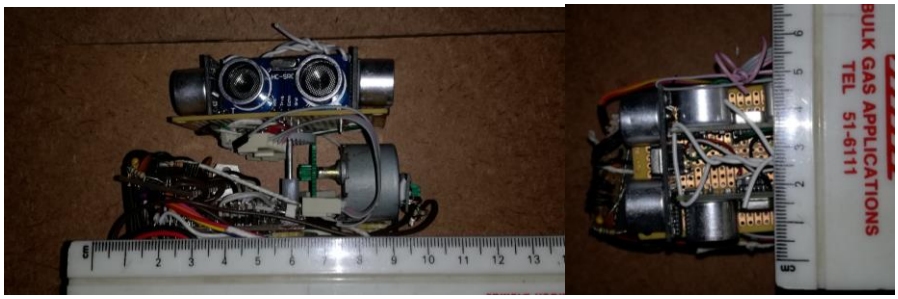


Figure 85: Length and width of the mobile robot

The mobile robot is less then 20cm x 20cm.

3.2.8 Experiment 8: Qualification test 8 The mobile robot must run though entire maze in under 6 minutes.

3.2.9 Experiment 9: Qualification test 9 The remote, must be wireless and work within a radius of 10 m.

3.2.9.1 Qualification tests

The objective of this test is to find the distance that the remote control no longer works with the mobile robot. The mobile robot must be wireless and work within a radius of 10m.

Equipment:
- remote control built
- mobile robot built
- measuring tape

Experimental procedure:
The mobile robot and remote control was turned on.
The mobile robot and remote control was placed 5cm from each other
A command was sent to the mobile robot
If the mobile robot responds correctly the distance between the mobile robot and the remote was increased by 30cm and retested.
The distance that the mobile robot no longer responds to the mobile robot was recorded.

3.2.9.2 Results and observations

The distance that the mobile robot no longer responds to the remote control is 30cm

3.2.10 Experiment 10: Qualification test 10 The sensors must detect the distance of a wall with an error of less than 14%.

3.2.10.1 Qualification tests

The objective of this test was to find the error that the sensors create at different wall angles. The sensors must detect the distance of the walls with an error of less than 14%

Equipment:
- the walls of the maze
- the mobile robot

Experimental procedure:
The mobile robot was placed in the maze with the walls of the maze placed parallel to the mobile robot.
The mobile robots sensor system was run and the distances are recorded.
These distances are compared with theoretically calculated measurements.

3.2.10.2 Results and observations

angle	Distance measured	Actual distance	error
-15	**140+154=297**	**289.9**	**2.44**
0	**135+147=282**	**280**	**0.71**
15	**118+153=271**	**289.9**	**6.52**
-15	**138+156=294**	**289.9**	**1.14**
0	**132+147=279**	**280**	**0.36**
-15	**119+153=272**	**289.9**	**6.17**
-30	**150+172=322**	**323.3**	**0.42**
-15	**141+151=292**	**289.9**	**0.724**
0	**138+152=290**	**280**	**3.57**
15	**144+164=308**	**289.9**	**6.24**
30	**977+1197=2174**	**323.3**	**572.4**
-30	**162+1196=1358**	**323.3**	**>100**
-15	**146+160=306**	**289.9**	**5.55**
0	**138+152=290**	**280**	**3.57**
15	**139+153=292**	**289.9**	**0.72**
30	**995+1197=2192**	**323.3**	**>100**

The maximum angle that the ultrasonic sensors can receive an accurate reading was 30°. As the sensors create such larger errors for angles greater than 30°, these readings were ignored. The error of the data used was therefore less than 14%.

3.2.11 Experiment 11: Qualification test 11 The mobile robot, must drive at a speed of greater than 5 cm/s and slow down to stop.

3.2.11.1 Qualification tests

The objective of this test was to find the travel speed of the mobile robot. The mobile robot must drive at a speed of greater than 5 cm/s and slow down to stop.

Equipment:
- the mobile robot
- ruler
- timer

Experimental procedure:
The mobile robot was run in the forward mode
The mobile robot was measured for distance over a set time.

3.2.11.2 Results and observations

The mobile robot moved 41.5cm in 10seconds, 4.1cm/s.
The mobile robot rans at a maximum of 4cm/s. When the speed increased by changing the delay between commands sent to the motors, the motors no longer had enough torque to move the mobile robot.

3.2.12 Experiment 12: Qualification test 12 The mobile robot must turn at a speed of at least 45° in 10 seconds

3.2.13 Experiment 13: Qualification test 13 The data storage unit needs to be at least 10 Mb in order to store all the relevant data.

3.2.13.1 Qualification tests

The data storage unit needs to be at least 10 Mb in order to store all the relevant data.

No extra data storage unit was used as the storage on the DSPIC is 16 Mb which was large enough to store all the relevant data.

3.2.14 Experiment 14: Qualification test 14 The mobile robot must send the stored data to a PC though a cable within 30 seconds.

3.2.14.1 Qualification tests

The object of this test is to find the speed that the mobile robot sends all the information of the walls to the PC for display. The mobile robot must send the stored data to a PC though a cable within 30 seconds.

Equipment:
-PC
-Timer
-Mobile robot

-RS232 to USB cable

Experimental procedure:
The mobile robot sent 1344 zeros to the PC through the RS232 to USB cable.
The time for all the information to be received by the program Putty was recorded.

Theoretical analysis

48 distances x 2 coordinates each x 7 positions x 4 positions = 2688 coordinates
Each coordinate becomes 4 ASCII bits to send = 10752 bits
UART set up with a baud rate of 9600 bits/second

= 1.12seconds to send all of the data to Putty.

3.2.14.2 Results and observations

The time that all of the data was sent to Putty was less than 30 seconds.

3.2.15 Experiment 15: Qualification test 15 The remote must be battery driven for at least to 20 minutes.

3.2.15.1 Qualification tests

The objective of this test is to find the time that the battery system of the remote will run for. The remote must be battery driven for at least 20 minutes.

Equipment:
-remote control
-timer

Experimental procedure:
The remote is used over a 20min period

3.2.15.2 Results and observations

The remotes' battery system did not deteriorate over a 20 minute period.

3.2.16 Experiment 16: Qualification test 16 The remote must have five movement options: forward, backward, rotate left, rotate right and stop.

3.2.16.1 Qualification tests

The objective of this test is to determine if the remote control allowed for five inputs. The remote must have five movement options: forward, backward, rotate left, rotate right and stop.

Equipment:
-The remote

Experimental procedure:

Observe that the remote allows for five inputs

3.2.16.2 Results and observations

The remote allowed for five inputs, four push buttons and one switch button. The stop button was incorporated into the switch and the operation of the push buttons. When the buttons was pressed they allowed the mobile robot to respond to the commands and when the push buttons were released the mobile robot stops.

3.2.17 Experiment 17: Qualification test 17 The height of the walls of the Maze must be less than 15 cm, and the walls must be thinner than 1 cm.

3.2.17.1 Qualification tests
The objective of this test is to find the height, and thickness of the maze walls. The height of the maze walls must be less than 15 cm, and the walls must be thinner than 1 cm.

Equipment:
- Ruler
- Walls of the maze

Experimental procedure:
The thickness of the walls of the maze was measured
The height of the walls was measured.

3.2.17.2 Results and observations

The height of the walls was measured at 14cm
The thickness of the walls was measured at 0.6cm

The height of the mobile robot was 12cm, which was less then 14cm in order for the sensor system to read the detail of the walls.

Figure 89: height of the mobile robot

4. Discussion

4.1 Interpretation of results

Mission critical results

Display

The display was able to create the detail of some wall configurations within the given 15% requirement. However the limit of the sensors system was shown with sharp angled walls and corners that created larger errors in the display and therefore the detail of these configurations were lost. This is an undesired result. Increasing the number of samples taken in one scan allowed for more detail of the walls to be displayed. Increasing the number of samples taken however meant that the mobile robot would run thought the maze longer and therefore put pressure on the battery system. A larger battery system would create a need for motors with more torque and therefore larger motors that would increase the size of the mobile robot and the size of the maze. The problem with the sharp angles and corners of the maze is that it cannot be compensated for by increasing the number of samples. A different sensor system would be required to decrease these errors. The current sensor system was tested and large errors were rejected to increase the accuracy of the display. Walls with angles of more than 30° created large errors and were rejected.

Maze

The size of the maze is required to be at least 10 times larger than the mobile robot, this specification was made to ensure that a large maze was used and tested. The mobile robot was therefore made compact to correspondingly keep the size of the maze compact in order to satisfy other requirements, such as time through the maze.

Remote control

The mobile robot responded correctly to the remote control instructions within the one second requirement, this specification allows for the mobile robot to be changed from an automatic mode to a manual mode. In a self-driving vehicle, the vehicle would be required to run automatically and to have the option to be controlled manually by a user. The response time of the mobile robot is required to be as small as possible, as with a self-driving vehicle. The vehicle would need to respond to given commands with little delay for emergency situations.

Movement

The mobile robot has to move around the walls within the minimum distance of 3cm. This test showed that the mobile robot failed this specification however the 3cm specification was made as an average ultrasonic sensor as a minimum range from 3cm and therefore the detail of the walls for the display would be incorrect and create errors. The ultrasonic sensors used in this project had a minimum distance of 2cm and therefore allowed the mobile robot to move around the walls within 2cm and not minimize errors in the display.

The functional specification that were within specification

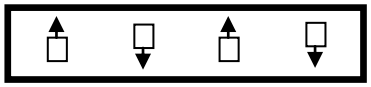
-The mobile robot sent the data to a PC through a cable within the required 30seconds. The result made the display system work much faster than expected.
-The mobile robots dimensions were found to be less than the required 20cm x 20cm, this allowed the mobile robot and the maze to be kept small.
-The remotes battery system allowed for the mobile robot to be driven through the maze, and worked for more than the required 20 minutes. The remote also allowed for the necessary forward, backward, rotate left and right instruction to be sent. The stop instruction is built into the push button system where the mobile robot stopped when no instructions were sent.
-The height of the walls, 14cm were less than the required 15cm and this restricted the height of the mobile robot, which was 12cm, in order to obtain the detail of the maze walls.

The functional specification that were not within specification

The mobile robot did not run through the entire maze in under 6 minutes. This requirement is failed as the motors could not provide enough torque to increase the speed of the mobile robot and to increase the accuracy of the display, (five distances were taken at each angle to create an average). Therefore, the speed requirement was failed. To pass the speed and time requirements the mobile robot needed more expensive motors that were small in size with higher torque. The servo motors allowed for an increase in speed and would have allowed the mobile robot to meet the speed and time requirements. However, the accuracy of the movements would have been reduced and therefore the accuracy of the display, shortest route and movement of the mobile robot are reduces. The display, shortest route and completion of the maze accurately were chosen over the speed and time requirements. The remote control does not work with a range on 10m, chosen to increase the degree of freedom of the user, but works in a range of 30cm.

4.2 Aspects to be improved

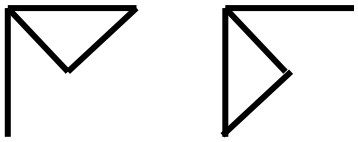
<u>Movement</u>
The movement of the mobile robot was improved from the first design by changing from the servo motors to the stepper motors. However, more proportional control would have created more accurate movements and therefore a more accurate display and shortest route calculation.
The movement only allowed for the mobile robot to run through the maze forward or backward mode in order to keep the code shorter and simpler. This however did not produce the most efficient movement. The following sketch shows the limitations of the mobile robot.



The mobile robot moved through this configuration by first moving left and turning to face backward then left facing forward as shown in the figure. The mobile robot therefore was able to move through this configuration and display but this configuration this was not the most efficient movement. The algorithm would need to

be made longer and more complex to allow for the mobile robot to move more efficiently around this configuration.

The mobile robot struggled to respond to the following wall configurations that are in corners of the maze.



To allow the mobile robot to respond to these configurations a different sensors would need to be used.

Display

The ultrasonic sensors responded well to most of the wall configurations. The sensor system had the most problems with corners or dead ends of the maze as this created sharp angles or walls that cannot be detected by the mobile robot (as shown in the results section of this report). In order to decrease theses errors a different and more expensive sensors system would be required, such as image processing, with cameras. The mobile robot would need to make smaller movements around the maze and take more samples, this would increase the time that the mobile robot would need to move around the maze and the storage required. The following shows how the mobile robot movements could have been increased to improve the detail of the maze walls.

The time that the mobile robot rans through the maze could have been decreased by using more expensive, smaller and powerful stepper motors, and decreasing the number of samples taken by the sensor system. However, this would have decreased the accuracy of the display. The accuracy of the display was prioritised over the travel time through the maze.

4.3 Strong points

The maze design worked well in being able to change the maze configurations and allowed for more configurations then originally designed.
The size of the mobile robot was made smaller than expected resulting in a compact maze design.

4.4 Under which circumstances will the current system fail?

The system will fail when different wall configurations are used such replacing a half wall with a two thirds size wall. The mobile robot will read this wall as a dead end and not move around it. It will also not included for the shortest route calculation. The display however, will show that there is a gap in the walls.

The algorithm was designed to be easily changed to cater for different maze sizes. For example, if the maze was made to be 2m x2m, making each block 50cm x 50cm, the mobile robot would only need to change its reference values. The mobile robots current algorithm however, would not work with a maze changed to 2m x 2m.

The mobile robot had difficulty moving around the following wall configurations.

The mobile robot cannot recover form a wrong decision being made and this was critical to the system. The movement of the mobile robot system needed to be as accurate as possible, as well as the sensors system in terms of its distance readings and rotation.

The mobile robot needed to start in an empty block, in the middle, parallel to the outer walls else the mobile robot would make incorrect judgements of the walls around itself as it is programs to know that it starts in a centre of a block. This will create an error of the judgements of the walls around itself, the display and shortest route.

4.5 Design ergonomics

The remote has four push buttons and a switch that are linked to LEDs to show the user if the switch is on or off and if the buttons are pressed. The remote was built to be hand held and light.
The mobile robot was compact in design and allows for a rectangular case to be placed around the processing unit to be more aesthetically pleasing.
The maze was designed for the base to be split into two parts, for easy storage and transport, and all the walls and clips could also be removed easily.
The battery system was designed to be recharged while on the mobile robot and the remote.

4.6 Health and safety aspects of the design

The remote control did not get hot and when placed in a case, the circuitry and wireless would be protected from the user. The mobile robot was designed to run automatically with little user contact, however the circuitry can be covered with a case.
There are no high voltages or high current used in this project to cause harm to the user.

4.7 Social and legal impact and benefits of the design

This project is part of the future car series project. When this software is integrated into a self-driving vehicle, the vehicle will be able to drive through a town or city automatically. Possible benefits: to decrease traffic congestion or to avoid traffic congestion decreasing the time wasted in traffic. An automatic car will allow for an inebriated person to be driven home without directly driving the vehicle.

4.8 Environmental impact and benefits of the design

The possible benefits from the technology of the future car series include reductions in fuel consumption and emissions due to a reduction in distance and travelling time by finding the shortest route around traffic congestions and by avoiding busy routes and accidents zones.
For this project specifically, the mobile robot and the remote used a rechargeable battery system for power. Rechargeable batteries can be reused and therefore batteries are not thrown away every time the mobile robot runs when the batteries are flat.
The maze was created from hardwood and therefore is not environmentally friendly as a tree needed to be chopped down to produce the hardwood. However this wood can be recycled.

5. Conclusion

5.1 Summary of the work

The mobile robot

The mobile robot was split into three main hardware components: the sensor system, the processing system and the motor system. The sensor systems comprised of four ultrasonic sensors that were placed at 90° on a platform. The platform allows the sensor system to be rotated in relation to the rest of the mobile robot to obtain a 360° view of the walls of the maze. The platform was controlled by a stepper motor below. The motor system was comprised of two stepper motors, two ordinary wheels and one omnidirectional wheel. The motors and wheel placement were optimized for the most control over the movement of the mobile robot. The main processing system was a DSPIC. The DSPIC controlled the algorithms for the sensors, the motors, the movement, the shortest route algorithm, the display, the communication between the remote and the communication between the PC.

The maze

The maze was built to be ten times larger than the mobile robot and to allow for complex configurations that could be changed easily. The maze and walls were built from hardboard. The walls could be inserted into the maze through a bulldog clip system. The clip is screwed into the board and the walls of the maze are held up by the clip.

The remote control

The remote control was built to allow for four push buttons to send information of the movement of the mobile robot. A wireless communication device is programmed to send the information from the push buttons to the mobile robot. The push buttons are connected to a switch that will allow for the buttons to be deactivated.

The display

The sensor system obtains the distances of the walls around the mobile robot. This information is processed and normalized to the position and orientation of the mobile robot in the maze. The distances are converted into coordinates that are sent via UART through and RS232 to USB cable to the PC. The PC inputs the data into a text file and Dev–c++ runs an algorithm to display the walls of the maze.

The movement algorithm

The movement algorithm was created to allow the mobile robot to run around the walls of the maze and to run through the whole maze without running a route unnecessarily.

The shortest route algorithm

The shortest route algorithm was created to calculate the shortest route through the maze and displays the shortest route.

5.2 Summary of the observations and findings

- The display of the system was only accurate when the wall configurations did not use sharp angles.
- The mobile robot responded to the instructions of the remote control however the range that the remote worked was much smaller than required.
- The size of the maze was found to be ten times larger than the mobile robot.
- The mobile robot moved around the wall configurations with a minimum distance of 2cm.
- The mobile robots movement system was slower than required as the motors were found to not have the required torque to move the mobile robot at a faster speed.

5.3 Suggestions for future work

- The movement and shortest route algorithms should be tested with a vehicle, and road situations.
- The mobile robot should be integrated with other systems to form a self-driving vehicle.
- The accuracy of the display of the maze could be improved by using a more complex sensor system or image processing.

6. References

[1] M. A. Makhal, “Path Planning through Maze Routing for Mobile Robot with nonholonomic constraints,” *IEEE,* vol. 1, pp. 325-331, 2012.

[2] J. a. Z. S.H. Cao Hongyu, “Multi-ultrasonic-sensor Grid Map Building Based on D-S Evidence Theory,” *IEEE,* vol. 1, pp. 910-915, 2015.

[3] D. S. K.Kalmegh, “Obstacle avoidance for a mobile exploration robot using a single ultrasonic range sensor,” *IEEE,* vol. 1, pp. 8-11, 2010.

[4] M. N. H. I. M. N. H. I. a. H. H. S. Yano, “A study of maze searching with multiple robots system,” *IEEE,* vol. 1, pp. 207-211, 1996.

[5] J. S. Q. G. H. Dang, “An Efficient Algorithm for Robot Maze-Solving,” *IEEE,* vol. 1, pp. 78-82, 2010.

[6] D. M. U. H. F. Ahsan, “Seeker: Autonomous Maze-Navigating and ball-potting robot,” *IEEE,* vol. 1, pp. 52-57, 2015.

[7] V. T. A. J. G. G. O. Kathe, “Maze Solving Robot using Image Processing,” *IEEE,* vol. 1, pp. 1-5, 2015.

[8] G. S. M. R. J. A. R. A. F. M. S. A. A. M. Khan, “Design and Implementation of a Robot for Maze Solving with turning indicators using PID controller,” *IEEE,* vol. 1, pp. 1-6, 2013.

[9] A. Dzozdek, Data structures and algorithms in java, Fourth Edition.

[10] G. RS, “Stepper motors and drivers, what is full step, half step and microstepping?,” Design spark, 7 6 2015. [Online]. Available: https://www.rs-online.com/designspark/stepper-motors-and-drives-what-is-full-step-half-step-and-microstepping. [Accessed 7 11 2016].

[11] F. Reed, “How do servo motors work,” [Online]. Available: http://www.jameco.com/jameco/workshop/howitworks/how-servo-motors-work.html. [Accessed 10 10 2016].

[12] G. Lazaridis, “How sepper motors work,” 7 5 2010. [Online]. Available: http://pcbheaven.com/wikipages/How_Stepper_Motors_Work/. [Accessed 11 10 2016].

[13] R. Ron, “4 wheeled robot design basics and challenges,” 27 2 2014. [Online]. Available: http://www.rakeshmondal.info/4-Wheel-Drive-Robot-Design. [Accessed 11 10 2016].

[14] “Univeral robotics - wheel control theory,” [Online]. Available: http://www.robotplatform.com/knowledge/Classification_of_Robots/wheel_control_theory.html. [Accessed 22 10 2016].

[15] “How does an ultrasonic sensor work?,” GK-12 Program, Computational Neurobilogy Center College of Engineering University of Missouri, [Online]. Available: https://www.teachengineering.org/lessons/view/umo_sensorswork_lesson06. [Accessed 10 10 2016].

[16] “Ms. Ledoux's class,” 2016. [Online]. Available: http://msledoux2012.weebly.com/bess-background-information.html. [Accessed 11 10 2016].

[17] [Online]. Available: http://www.rason.org/Projects/regulator/Schemvr.gif. [Accessed 11 10 2016].

[18] [Online]. Available: http://www.fivestarstationery.com/images/product/aplus%2032mm%20binder%20clip.jpg. [Accessed 12 10 2016].

[19] Z. D. BP Lathi, Modern digital and analogue communication system.

[20] “ASCII table and description,” [Online]. Available: http://www.asciitable.com/. [Accessed 16 10 2016].

[21] Shodor, “Simple statistics,” Department of Chemistry, the university of North Carolinaat Chapel Hill, [Online]. Available: https://www.shodor.org/unchem-old/math/stats/index.html. [Accessed 5 11 2016].

7. APPENDIX

APPENDIX 1: Sensor code

```
void sensor_0(void)
{
    counterb=0;  //clear all values
    Dist0=0;
    dist_0 =0;
    while(counterb<2)       //average two readings for accuracy
    {
    counter=0;
    countA = 0;                 //send pulse
    PORTBbits.RB3=1;            //RB3 - output
    __delay32(4000);            //100us =(4000/40000000)
    PORTBbits.RB3=0;
        while(countA<200)
        {
        while(PORTAbits.RA0 ==1&& counter <20000)               //RA0 - input
                {counter++; }
                __delay32(1);
                  countA++;}
        counterb++;
        Dist0 = (counter * 0.44 )/10 +38 ;
        dist_0 = (dist_0 + Dist0)/counterb;   // +38 ; //distance into mm
                                              //moved to reference.
    }
    if(dist_0==38)  //no return pulse
    {dist_0=800;}
}
```

APPENDIX 2: Sensor system code

```
void scan(void)
{
    scan_count =0;
    degree =0;
    move=0;
    second=0;
    while(scan_count <23)
    {
    sensor_0();     //run each sensor for distance
    sensor_1();
    sensor_2();
    sensor_3();

    if(second==0)                   //every second distance is saved
    {                               //(every 15 degrees)
    movement[0][move]=dist_0;       //save diatances into array [0][x]
    movement[1][move]=dist_1;       //move 0-2 (0,1,2)
    movement[2][move]=dist_2;
    movement[3][move]=dist_3;
    move++;
    second=1;
    }
    else if(second==1)
    {second=0;}

    scan_count=scan_count+2;
    if(scan_count==24)
    {rotate(132);            //rotate back to original
    }else
    {rotate_back(12);      //rotate 7.5 degrees
    } }
}
```

APPENDIX 3: Display code

```
    if(turn ==0 || turn ==3)        //if forward or backward co ordinates
    {
        if(turn ==3)                //if backwards swap x and y distances for display
        {
            dist_00 = dist_3;
            dist_11 = dist_2;
            dist_22 = dist_1;
            dist_33 = dist_0;
        }else if(turn ==0)
        {
           dist_00 = dist_0;
            dist_11 = dist_1;
            dist_22 = dist_2;
            dist_33 = dist_3;
        }
    if(dist_00 >180 || dist_00 <0)      //if distances out of wall range (ignore)
    {dist_00=0;}
    if(dist_11 >180 || dist_11 <0)
    {dist_11=0;}
    if(dist_22 >180 || dist_22 <0)
    {dist_22=0;}
    if(dist_33 >180 || dist_33 <0)
    {dist_33=0;}
      //convert to coord
      //scale the coordinated for display
x0_coord= (dist_00*cos((2*pi/3)-degree))/3.2;    //x0    0-1-2
y0_coord= (dist_00*sin((2*pi/3)-degree))/3.2;    //y0
x1_coord= (dist_11*cos((7*pi/6)-degree))/3.2;     //x1    3-4-5
y1_coord= (dist_11*sin((7*pi/6)-degree))/3.2;     //y1
x2_coord= (dist_22*cos((pi/6)-degree))/3.2;    //x2    6-7-8
y2_coord= (dist_22*sin((pi/6)-degree))/3.2;    //y2
x3_coord= (dist_33*cos((5*pi/3)-degree))/3.2;   //x3    9-10-11
y3_coord= (dist_33*sin((5*pi/3)-degree))/3.2;   //y3

coord[scan_count+24*counterc][0] = floor(x0_coord);               //x
coord[scan_count+24*counterc][1] = floor(x1_coord);
coord[scan_count+24*counterc][2] = floor(x2_coord);
coord[scan_count+24*counterc][3] = floor(x3_coord);
coord[scan_count+1+24*counterc][0] = -floor(y0_coord);             //y
coord[scan_count+1+24*counterc][1] = -floor(y1_coord);
coord[scan_count+1+24*counterc][2] = -floor(y2_coord);
coord[scan_count+1+24*counterc][3] = -floor(y3_coord);
}
degree= degree+(pi/24);
```

APPENDIX 4: Normalize display coordinated

```
if (turn==0||turn==3)
{
count_coord=0;
count_coord2=0;
while(count_coord2<12)
{
while(count_coord<4)
{
    if(coord[24*counterc+2*count_coord2][count_coord]!=0)
    {
    coord[24*counterc+2*count_coord2][count_coord]=(coord[24*counterc+2*count_coord2][count_coord]+ref_x);
    }
    if(coord[24*counterc+2*count_coord2+1][count_coord]!=0)
    {
    coord[24*counterc+2*count_coord2+1][count_coord]=(coord[24*counterc+2*count_coord2+1][count_coord]+ref_y);
    }
    count_coord++;
}
count_coord2++;
count_coord=0;
}
}
```

APPENDIX 5: UART for display

```
void __attribute__((__interrupt__)) _U1TXInterrupt(void)
{//if interrput on sending data clear flag and send coordinates
IFS0bits.U1TXIF = 0; // Clear TX Interrupt flag
if (uart_count<1345)
{
    uart_count=uart_count+1;
    if(three==0)
{
    if(q==4)
    { q=1;
      p++;
    }else
    { q++; }

each[3]=45;                             //send -
each[2]= floor(coord[p][q-1]/100) +48;          //hunderds,tens ones then into ascii
each[1]= floor(((coord[p][q-1]-(100*(each[2]-48)))/10)) +48;
each[0]=(coord[p][q-1]-((100*(each[2]-48))+(10*(each[1]-48))));

if(each[0]==0 ||each[0] == 1||each[0] == 2)
{each[0]=1;}
if(each[0]==3 ||each[0] == 4||each[0] == 5)
{each[0]=4;}
if(each[0]==6 ||each[0] == 7||each[0] == 8||each[0] == 9)
{each[0]=7;}
each[0]=each[0]+48;
three=5;
}

three--;
U1TXREG =each[three-1];
}
```

APPENDIX 6: UART for transmitter(remote)

```
void __attribute__((__interrupt__)) _U1TXInterrupt(void)
{
IFS0bits.U1TXIF = 0;
if(PORTAbits.RA0==1)
{
U1TXREG =48;
}
else if(PORTAbits.RA1==1)
{
U1TXREG =50;
}
else if(PORTAbits.RA2==1)
{
U1TXREG =52;
}
else if(PORTAbits.RA4==1)
{
U1TXREG =54;
}
else
{
    IFS0bits.U1TXIF = 1;
}
}
```

APPENDIX 7: UART for receiver(mobile robot)

```
void __attribute__((__interrupt__)) _U1RXInterrupt(void)
{
    temp= U1RXREG;
   if(temp==228)
    {
        PORTB=0x8000;
    }
    else if(temp==144)
    {
        PORTB=0x4000;
    }
    else if(temp==34)
    {
        PORTB=0x2000;
    }
    else if(temp==32)
    {
        PORTB=0x1000;
    }

    IFS0bits.U1RXIF = 0; // Clear TX Interrupt flag

}
```

APPENDIX 8: PWM calculations

The following shows the calculations used for the PWM used for the servo motors that were replaced.
PWMCON1 =0Xffff; pmw pairs are in complementary mode, does not make a difference
PTCKPS =3 which is a prescaler of 64

$$\frac{1}{pulse\ frequency} = Tcy * prescale * PTPER$$

$$20ms = \frac{1}{40Mhz} * 64 * PTPER$$

$$PTPER = 12500$$

$$\frac{1}{pulse\ width} = \frac{Tcy}{2} * prescale * PDC1$$

$$1ms = \frac{1}{2 * 40M} * 64 * PDC1$$

$$PDC1 = 1250$$

$$2ms = \frac{1}{2 * 40M} * 64 * PDC2$$

$$PDC2 = 2500$$

Part 5. Technical documentation

This main report is supplemented with technical documentation. This provides more detail on the software that was used in the experiments, including program listings, a user guide and circuit designs. This section appears on the CD that accompanies this report.

The CD is organized into the directories listed below.

Main report
Part 5: Technical documentation
Software
References
Datasheets
Author
Datasets/Raw
Datasets/Final

I, Greg Lavagna, have been allowed adequate time to read the project final report of Kirsty Jones carefully and to make corrections where necessary.

To the best of my knowledge, correct formatting, spelling, and grammar are used throughout the document.

Greg Lavagna
073 625 9773
BA(Hons): English
BA(Hons): Translation and Professional Writing
BA(Languages): Journalism (Univ. of Pretoria)
Ex-State President of the Republic of Kollegetehuis