Department of Computer Science and Engineering - University of New South Wales

Robot to the Rescue

ENGG1000

Edited By

Mandy Chen (z5205689)

Selina Chua (z5208109)

Alana Hua (z5209503)

Heeyeon Jeong (z5205445)

Quynh Boi Phan (z5205690)

Elizabeth Zhong (z5209166)

Date Submitted: 01 June 2018

Executive Summary

The report includes the processes involved in the development of the problem statement, the design process, the final performance and the results. The significance of developing various designs that satisfy the project requirements and the selection of the most ideal design is elaborated through the variations of prototypes that cumulated to a final design. Furthermore, the report evaluates the various methods of concept generation, improvements and decisions that contributed to the robot’s performance and concludes that successful design stems from careful preliminary planning and execution of the ideas.

Acknowledgements

The authors would like to thank the following people and groups who have contributed to the success of our project:

- Claude Sammut (Lecturer)
- Colm Flanagan (Tutor)
- Michael Schofield (Lecturer)

Table of Contents

1 Introduction

1.1 Purpose Statement

The purpose of this report is to provide the client with a clear image of what the rescue robot looks like and how effectively it operates in relation to both time and navigation. It also provides the client with several suggestions to both hardware and software that could improve the robot's overall performance and efficiency.

1.2 Project Background

This project is based on the fact that there are many situations that become too dangerous for emergency personnel to handle.

The scope of this project requires the design and building of an autonomous rescue robot that can navigate through hazardous terrain, locate the victim, and rescue them from such a situation. In this project, we use a maze as a simplified simulation of such an environment, and a brightly coloured can to represent the victim. The robot therefore must navigate through the maze, locate the can, and bring it safely back out within a given period of time.

1.3 Problem Statement

To design and construct an autonomous robot that can navigate through a maze to pick up its victim and return to the start within the given time limit. The robot must comply to standards of the robot's program and its material limitations.

1.4 Key Questions Explored

Several questions that were essential to this project arose after the creation of the problem statement, including:

1. What is the best method of navigation through a maze?
2. How can the victim be retrieved?
3. How can the victim be transported back to the beginning of the maze?
4. How can the efficiency of the robot be increased?
5. What is required in the software and hardware to achieve these goals?

1.5 Report Outline

This report contains the key concepts generated by the team in conjunction with detailed evaluations of each design in relation to the generated problem statement. Our design process is then summarised, which was used to generate the most effective design which was used to develop the final prototype. The performance of the prototype in the final testing has also been presented with recommendations for improvements that can be implemented to increase the effectiveness of the design.

2 Design Concepts & Considerations

2.1 Design Concepts

Several methods, including researching and brainstorming, were used to come up with a holistic solution that satisfied the problem statement. A morph chart[1] was then used to visually capture the several ways in which these objectives could be achieved, which were then narrowed down to the most efficient choice.

Hardware

Software

Claw:
Able to maintain a grip on the can by either lifting it off the ground or dragging it along as the robot moves.

Sensors:
To make navigation as efficient and effective as possible.
Ultrasonic - detect presence of walls to determine which direction to move.
Gyro sensor - senses robot's angular direction, calibrates it to move in a straight line.
Colour sensor - detects red colour of the can, triggering the claw motors.

Navigation:
Research into several maze algorithms and selected the most appropriate one.

Figure 2.1.1: Flowchart of Design Concepts

The final design concept for the rescue robot consisted of several elements:

2.2 Design Consideration

The rules and restrictions[2] briefly identified within the problem statement were reviewed and considered in the process of concept generation. From this, the team came up with several objectives for the robot.

- Be able to turn right and left at corners, and move forward otherwise,
- Be equipped with a strategy to exit the maze,
- Be aware of its surroundings (detect walls and detect can),
- Be small and robust (for effective navigation and avoiding collision with walls),
- Be able to bring the "victim" out of the maze.

2.2.1 Claw Designs

One of the objectives of the project was the ability to pick up a victim (brightly coloured can) and transport them out of the maze. To address the objective some designs were considered:

1. Finger Grab: When the robot detects its victim, the arm reaches out and grasps its victim by the side.

Advantages	Disadvantages
• Simple, easy to build. • Strong grip - the whole side of the can is held onto like a normal hand holding a can. • Gear system - gears easily tightens the grip between arms and can	• Arm can only grab the can if it is placed vertically upward without a system for the arm to rotate 360 degrees. • Fingers must be able to wrap around more than the diameter of the can.

[1] *See appendix A for Morph Chart*
[2] *See appendix B for Rules and Restrictions*

2. Clamp: When the robot detects its victim, the arm reaches out and grasps its victim by the side.
<u>NOTE</u>: The advantages and disadvantages are the same as the 3-finger grabber. The advantages/disadvantages listed underneath denotes those that were not listed in 3-finger grabber

Advantages	Disadvantages
● Compact	● Weaker grip - Only a certain area of the can is held; the rest of the can has no grip. ● Victim must be more than the diameter of the circle made inside the clamp.

3. Arcade Claw: When the robot detects its victim, the arm reaches out and grasps its victim by the top.

Advantages	Disadvantages
● The can can be displaced in any way and the claw should still be able to grab the can. ● The whole surface area of can is grasped onto	● Can will easily slip off if grip is not strong enough, can is too heavy or grabbed from top of can. ● System is complex - may be time consuming and not enough materials to build such system. ● Arm must be directly above can for best grip.

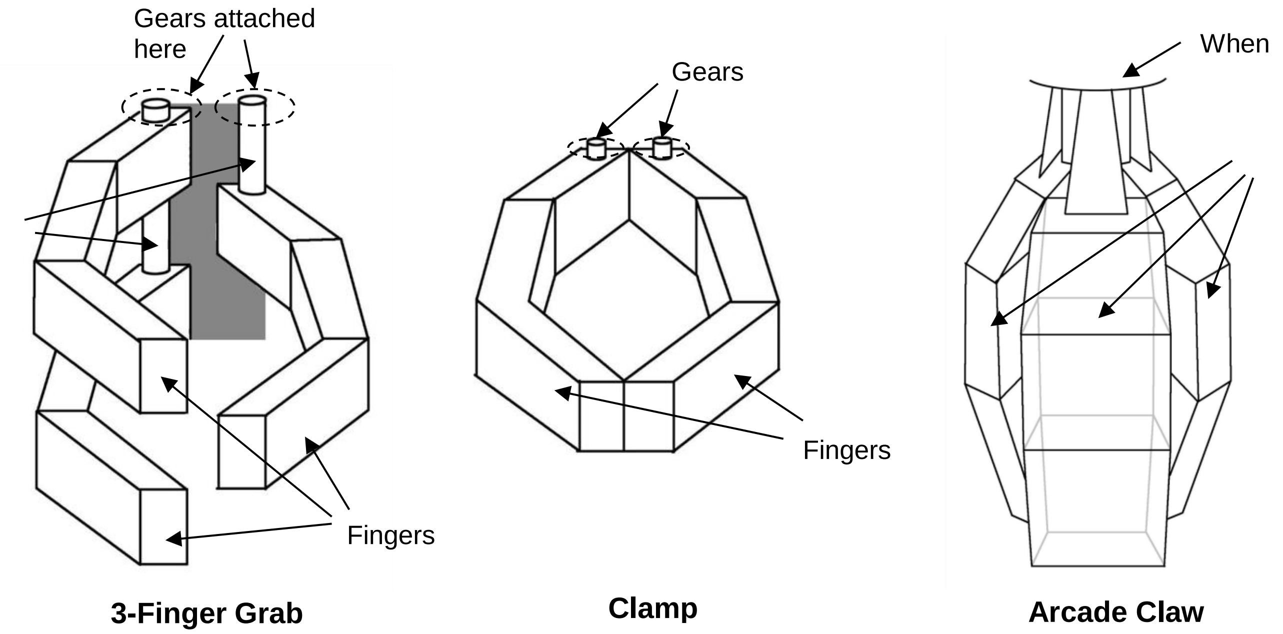


Figure 2.2.1: Labelled sketches of Claw Designs

2.2.2 Maze Navigation Tactics

Tactic	Description
Wall Follower	● Follows the wall until can and/or the exit is found. ● This rule is particularly useful for finding a path as it has a 100% success rate (as long as it does not start from *within* the maze as it's "hand" could be touching an island).
Tremaux	● When the robot enters the maze, it marks the path it follows. ● At junctions, it chooses random paths that have not been marked yet. ● If the robot has to return to a path that *has* been marked (i.e. a dead end), it will be marked twice - and suggests that it is not the most efficient path to the exit. ● The robot can exit the maze by following the path outlined by only one mark.

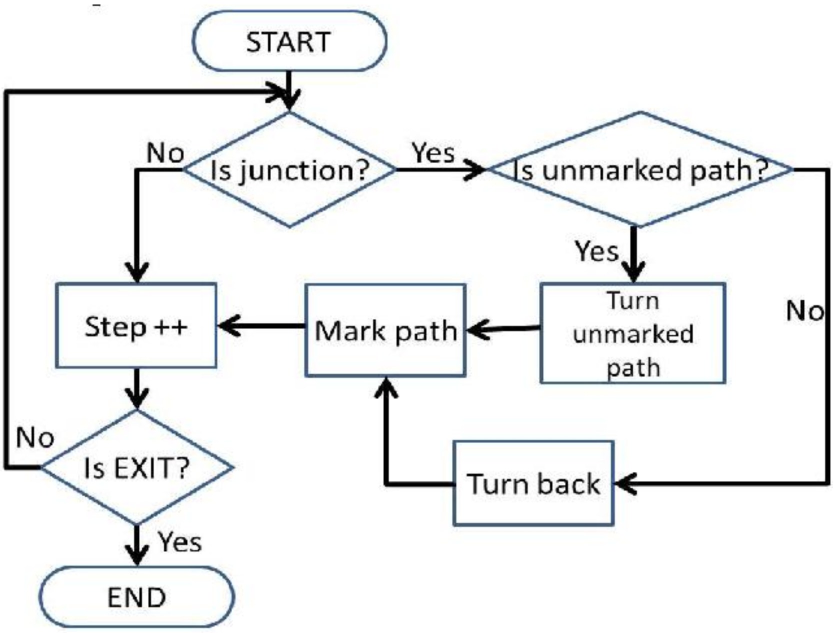


Figure 2.2.2: Flowchart representation of Tremaux Algorithm

2.3 Design Evaluation

Claw design

The 3-Finger Grab design was the overall choice for the claw. Both its simplicity and effectiveness were major factors: its implementation would be relatively easy and fast, and its interlocking mechanism would easily maximise grip. The way in which the gears are connected would also allow the claw to tighten around the can, and thus firmly grip it.

Maze solving algorithm

Although the wall follower was the most simplistic solution and had several holes (i.e. unable to escape a loop), the software team decided to implement it as it was extremely simple and would allow the robot to easily navigate through and out of the maze. Despite not being able to navigate through islands (walls that are not connected to the outer perimeter), it was still the most effective method to ensure that the robot could explore most of the maze.

Further improvements to software and the obvious loopholes within this strategy were later made.

3 Design Process

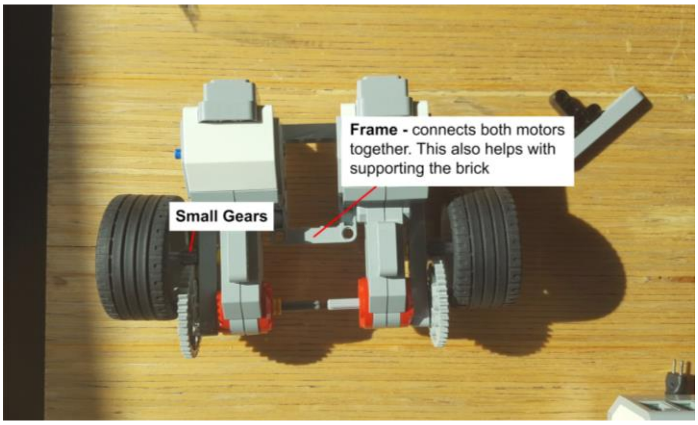
3.1 Hardware

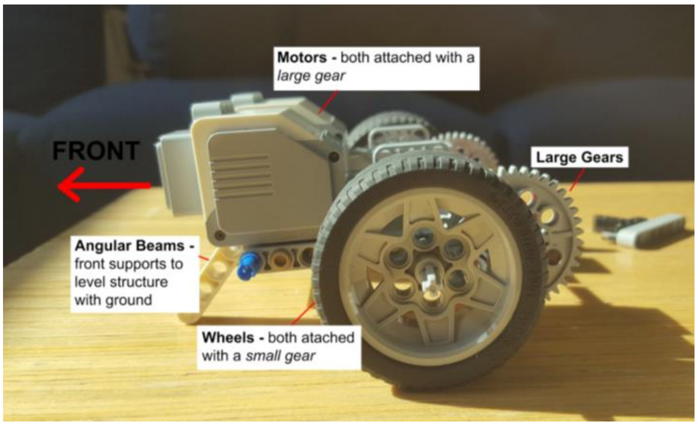
3.1.1 Base and Sensors Attachment

The sturdiness of the robot was the core focus of the hardware team when constructing the base of the robot as to ensure that its stability lasted throughout the whole maze. However, fixing the sensors into the robot such that functionality was not compensated was also a vital goal.

Design Prototype #1

Design Overview





Note: the gears were attached onto the motor and wheels as such to increase the robot's overall speed.

Figure 3.1.1: Labelled photos of Design Prototype #1

Evaluation

Due to an uneven distribution of weight (which was exaggerated when the brick was added), both the wheels and the motors slanted inward. The lack of frame strength affected the efficiency of the robot's motion and stability; and therefore, the team decided against this design.

Design Prototype #2[3]

Design Overview

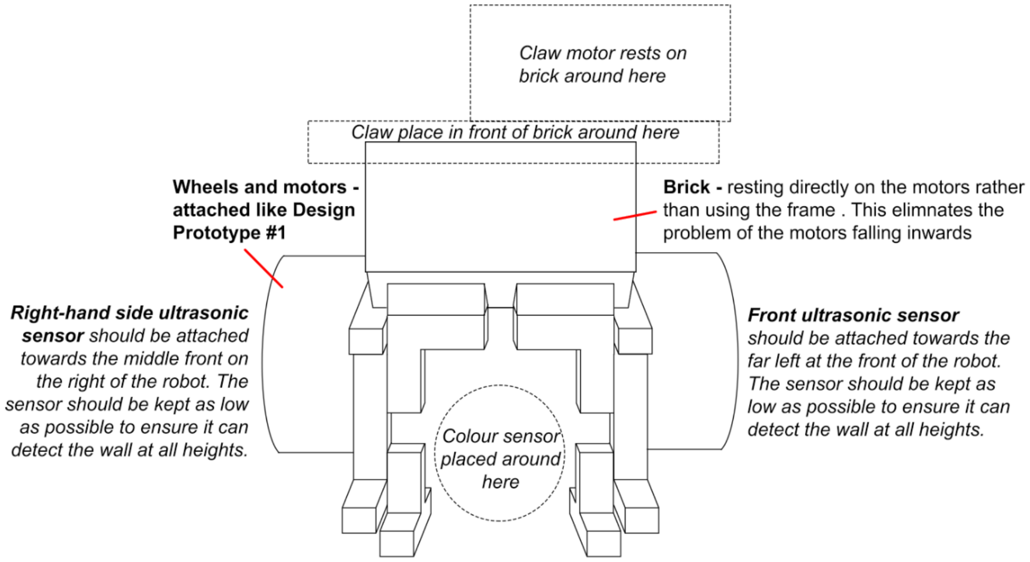


Figure 3.1.2: Labelled sketch of Design Prototype #2

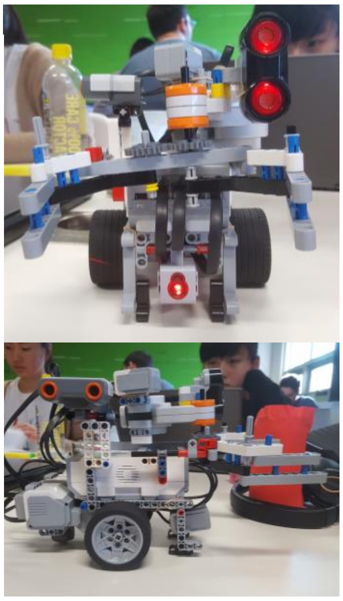


Figure 3.1.3: Photos of finalised design of Design Prototype #2

[3] See appendix C for photos before finalised design

Minor Adjustments for Final Design of Prototype #2

Structurally

- Gyro sensors added to ensure a perfect 90° turn. This was placed on top of the robot.
- Gears on the robot's sides were removed as it fell off too easily, which previously made the wheels prone to falling off at high speeds.
- Wheels were re-positioned to the middle of the robot; however, this made it unbalanced. T-shaped beams and a ball bearing were added as front and back supports respectively to balance the robot.

Sensors attachment

- Right-hand side ultrasonic sensor was placed towards middle back rather than middle front → allowed robot to stop turning right/left prematurely (because it sensed an empty space too early). It was also raised up as it would occasionally detect the can as a wall.

Evaluation

Although the robot was made sturdier, it now moved with too much force, causing the robot to constantly knock the can over when it was not centered with the claw. Despite an attempt to shift the claw lower, complications with space and placement for the claw motor led to this design being dismissed.

Design Prototype #3[4]

Design Overview

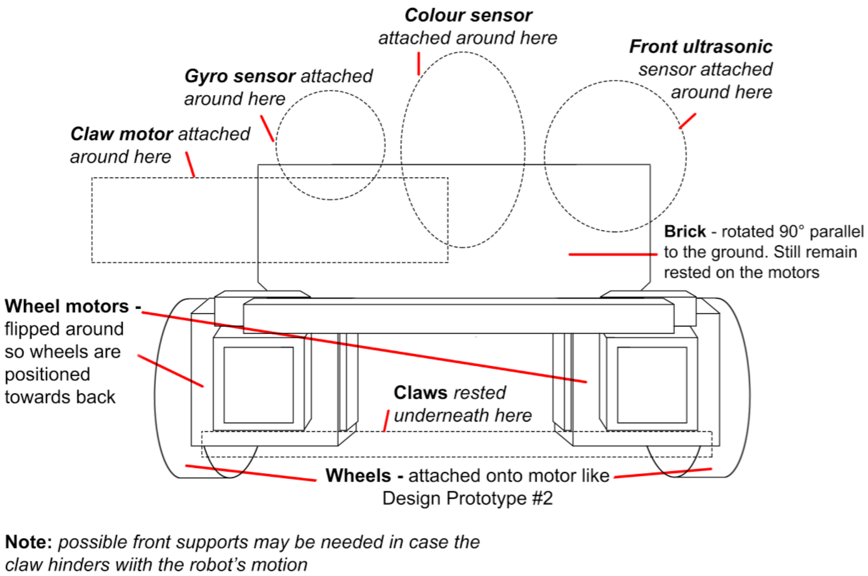


Figure 3.1.4: Labelled sketch of Design Prototype #3

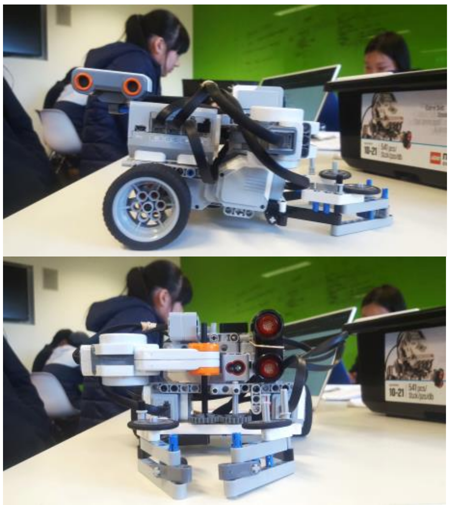


Figure 3.1.5: Photos of finalised design of Design Prototype #3

Note: *another ultrasonic sensor was attached towards the back on the right-hand side of the robot.*

Minor Adjustments for Final Design of Prototype #3

Sensors attachment

- Colour sensor was moved lower, centred in the middle of claw - allows it to detect can at all heights.

Evaluation

By improving on robustness and widening the claw, this last prototype allowed the robot to move at higher speeds without knocking the can over. The robot remained compact and thus would be able to easily maneuver through the maze without bumping into walls. This design became the final choice used at the performance stage.

[4] See appendix D for photos before finalised design

3.1.2 Claw

Throughout the development of the claw, its general structure was retained with only minor adjustments made. These adjustments were based around the use of rubber grips such that the can was held with a tight grip and would not be knocked over before it triggered the colour sensor.

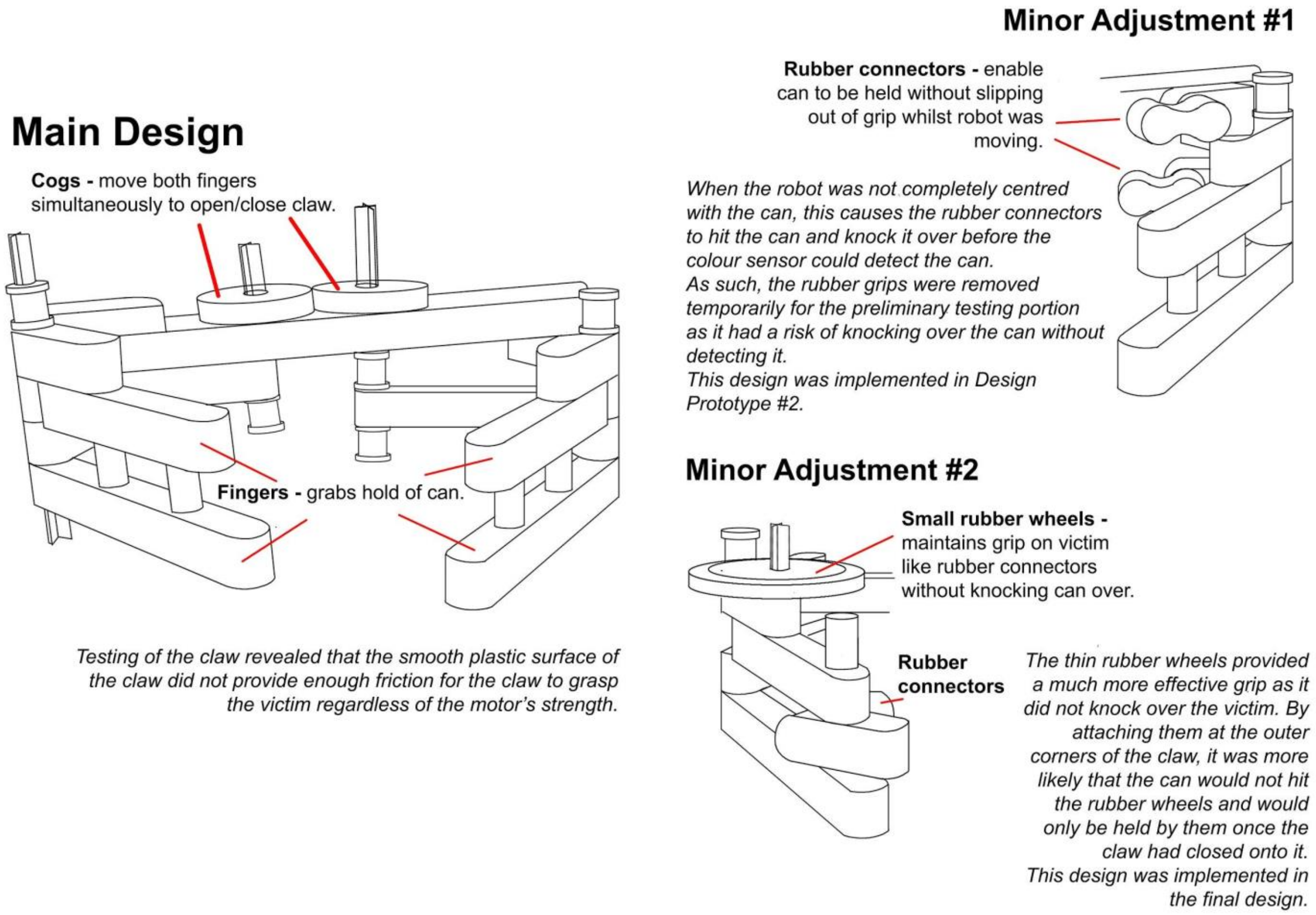


Figure 3.1.6: Labelled sketch and details of the Claw and its Minor Adjustments

3.2 Software

Software Prototype #1: Wall Follower (Right-hand rule)

Our initial implementation was the wall follower algorithm, which constantly followed the right wall.

Pros:

- Will always find a way out (if not placed in a loop)
- Relatively easy to implement
- Will eventually traverse through the entire maze (will always find the victim)

Cons:

- Time-consuming, especially in large and complex mazes
- Will remain in an infinite loop if the right-hand "touches" an island.
- Will re-enter all the dead ends when navigating out of the maze → could be extremely inefficient.

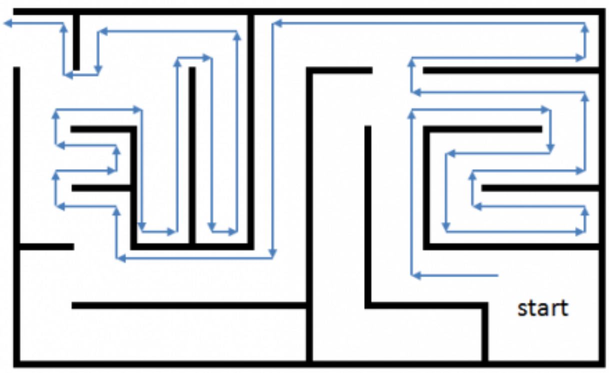
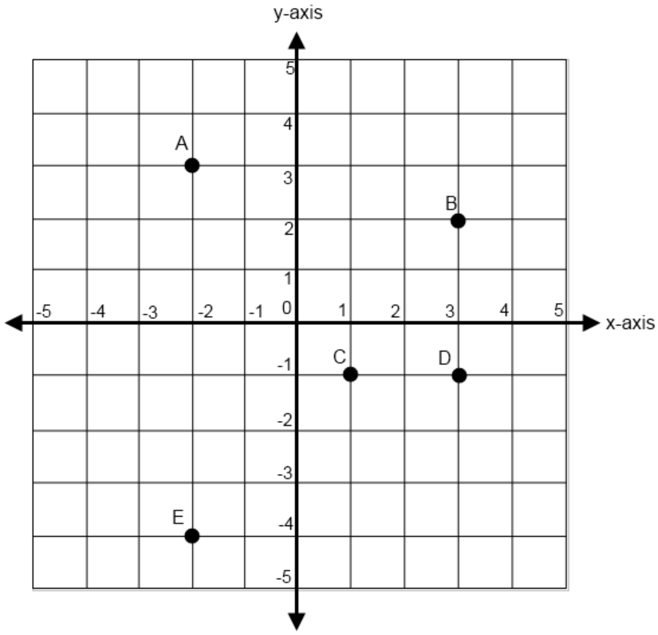


Figure 3.2.1: Visual representation of Wall-Follower

Evaluation: Although this ensures being able to exit the maze, the robot will be unable to find a can that is hidden within an island.

Software Prototype #2: Gridding

This method separated the maze into “grids” and utilised a coordinate system. We take *(0, 0)* as the starting position, front of this initial position as positive y, and right of initial position as positive x.



e.g. Moving forward 3 grids from starting position can be depicted as D → B.

Figure 3.2.2: Visual representation of Gridding

This strategy was still largely based on the right-hand rule but drew properties from Tremaux’s algorithm. Although the robot is programmed to follow the wall, it implements Tremaux’s double-mark concept such that the robot will not re-enter dead ends whilst still being able to successfully find the can and exit the maze. This also meant that the robot would not be stuck within an infinite loop as these junctions would have been double-marked.

Pros:

- Solves maze more efficiently (traces a path without entering previous dead ends)
- Can exit an infinite loop if it enters an island
- Able to pinpoint its exact position from the beginning (because of coordinates) and therefore can easily get back out of the maze

Cons:

- Relatively difficult to implement
- Maze might not be perfectly gridded

Evaluation: This strategy proves to be much more effective and versatile than the wall follower as it can adapt and overcome a series of situations (such as loops and not re-entering unwanted paths).

4 Performance

4.1 Testing Stage

Independent Testing

Testing the robot was achieved by testing the sensors and its overall performance within a maze.

Ultrasonic sensor

Depending on the ultrasonic sensor, the program will direct the robot to move as seen in *figure 4.1.1*. Note that the ultrasonic sensor detects if it is blocked by a wall if the ultrasonic sensor returns a value less than the minimum distance away from the wall.

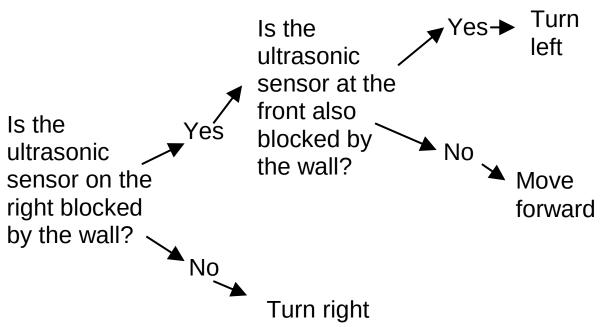


Figure 4.1.1: Flowchart diagram of program

Colour Sensor

The colour sensor test was made to analyse the different values returned when the colour sensor detects the red can and the white wall to determine the best value for detection of the can. This can be seen in *figure 4.1.2*.

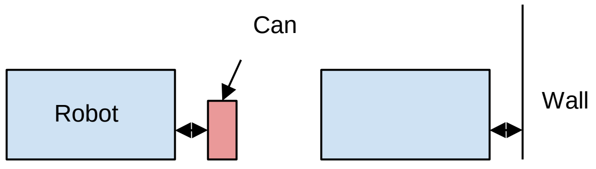


Figure 4.1.2: Visual representation of the colour test

Gyro sensor

To test if the gyro sensor was working properly or if any modification of the values in the program needed to be made, the following tests in *figure 4.1.3* were made.

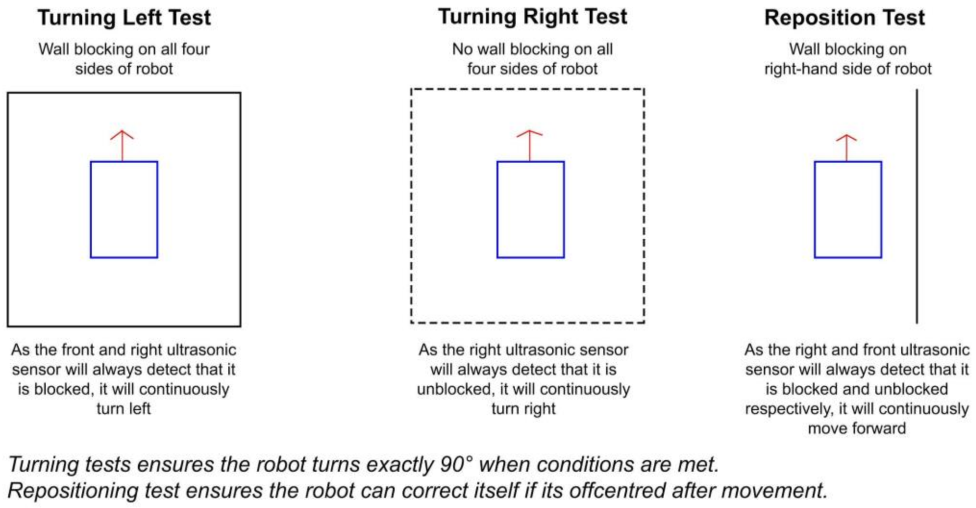


Figure 4.1.3: Visual representation and detailed explanation of gyro tests

The values retrieved from the testing stage was used to update the robot's program such that the robot's detection of its surroundings and sense of direction was optimised.

Preliminary Round

The functionality of the robot in navigating a maze was tested in a preliminary round before the performance test. In the preliminary testing stage, the robot navigated through the maze with minor issues. With the can inside the island, it was unlikely that the robot would enter the island to detect the can. Yet, the robot managed to rotate into the island as it detected the edge of one of its walls and retrieved the can. However, by this time, the time limit was exceeded and hence the robot received an 8/10. Its performance during the Preliminary round, suggested that the robot's speed was too slow.

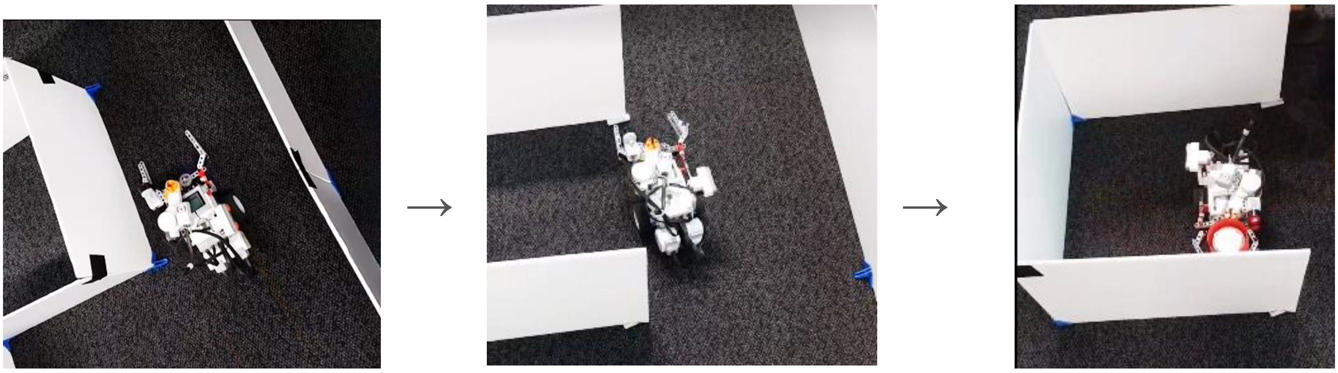


Figure 4.1.4: Photos of robot in preliminary maze

4.2 Final Performance[5]

The performance of the robot in the final testing was as follows:
Maze #1:
The robot was successful in locating the victim seamlessly without bumping into any walls. The victim was transported back to the beginning as predicted. It received full marks in all sections.

Maze #2:
Due to malfunction of the robot's gyro sensor, the robot was unable in detecting drift. As a result, the robot could not stay on course and crashed into the maze's walls several times as seen in Figure 4.2.1. This greatly impeded the time taken for the robot to reach the victim as the robot needed to be manually reposition to ensure that it was still able to navigate through the maze. This was emphasised when the robot was placed one grid away from the can and still veered off centre such that the can was not detected, causing it to lose one mark for finding the victim. Ultimately, the robot was not able to demonstrate that it could return to its starting position as it exceeded the time limit thus receiving 0/4 for that section.

Figure 4.2.1: Robot colliding into many walls in Maze #2

Maze #3:
The testing of the prototype in maze 3 revealed a flaw in the robot's loop detection system whereby the robot assumed that it was stuck in a loop but was unable to successfully exit. Consequently, the preliminary code which implemented a different loop detection system was run in replacement. However, due to the code being uncalibrated for the new hardware design of the robot, the robot, whilst managing to successfully reach and locate the victim, also veered off course like its performance in maze 2, hitting many walls, causing it to lose a mark in looking for the victim and running out of time to return to the beginning.

Overall:
In all mazes, the robot was able to generally locate the victim and detect and grasp the victim such that it would not be lost during its transportation back to the beginning.
Ultimately, the prototype's greatest weakness was that it was not able to move in complete grids and did not always detect itself moving off-center. If provided enough time, the team hypothesized that if the sensors were calibrated and more preliminary testing procedures were conducted, the robot would have been more successful in the final performance.

5 See appendix E for results of final testing

4.3 Improvements

Due to the lack of time available, the prototype can be still greatly improved. The team recommends the following improvements which can be implemented to increase the success of the robot.

HARDWARE

The rescue robot grasped and dragged the can along - something that would not be practical within a real-life emergency situation. Thus, improving the claw mechanism such that it could lift the victim would improve the effectiveness and practicality of the rescue robot.

Furthermore, the robot's movements during testing were largely inconsistent (e.g. did not turn 90º perfectly, tilting to the right when moving forward). This was due to the sensors moving when the program was started - managing the wires so they are not twisted would be able to solve this, as this will ensure accurate communication between the sensors and the program. The wires should also be neatly arranged to avoid it from hindering the robot's movement.

Moreover, due to the limited sensors available, the robot was not completely aware of its surroundings (e.g. ultrasonic sensor only covered a small portion of the robot's side). This caused the robot to detect a clear path too early/too late, and thus caused collisions with the walls. This could be improved by adding more sensors such that the robot has full awareness of its surroundings.

SOFTWARE

Because of constant hardware problems, it would be necessary for the client to calibrate the values after every round (e.g. gyro-sensor and ultrasonic sensor readings may vary). Moreover, because of time limitations, the team was not able to implement multi-threading for sensor readings. An implementation of this technique would rapidly improve robot performance as it would allow it to constantly sense for the can instead of being limited to sensing when it has moved one grid.

5 Conclusion

Overall, the rescue robot was generally capable of locating the victim using the wall follower strategy and then detect and grasp it. However, it was significantly weaker at moving in perfect grids which caused it to collide into many walls before finally reaching the location of the victim, slowing its progress, which resulted in it receiving 24/36 for the final mark. As such, future developments based on the above recommendations could be made so it can be more optimised for use in the real world.

6 Bibliography

avoid.py. (2018). UNSW Engineering.

Raor.ru. (2013). User Guide. [online] Available at: https://raor.ru/netcat_files/77/372/LME_EV3_UserGuide_US.pdf [Accessed 12 Mar. 2018].

Ward, N. (2016). EV3python. [online] EV3Python. Available at: https://sites.google.com/site/ev3python/ [Accessed 19 Mar. 2018].

7 Appendices

7.1 Appendix A: Morph Chart

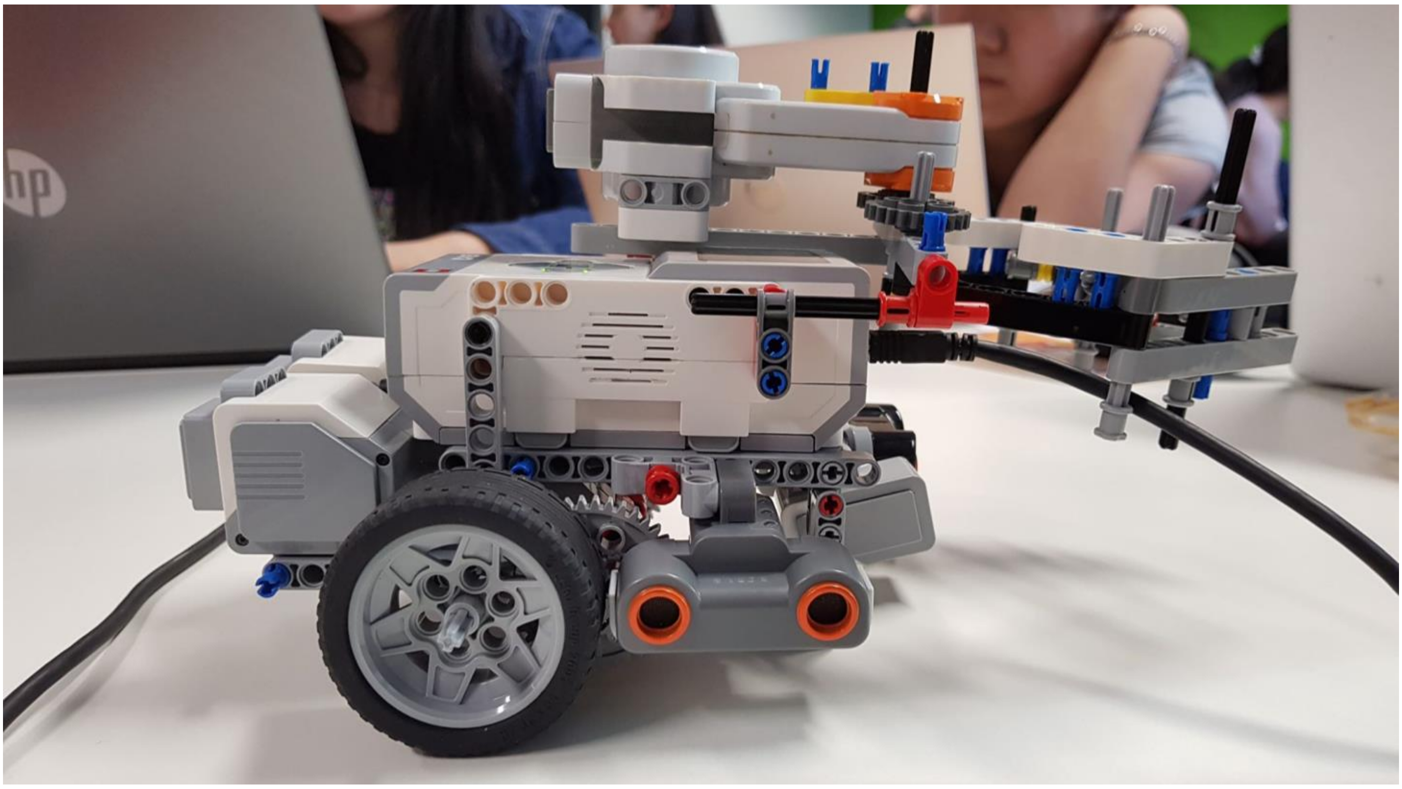
Attributes and Functions	**Option 1**	**Option 2**	**Option 3**
Navigate and exit maze	Trémaux's algorithm	Right-hand rule or left-hand rule	Pledge algorithm
Detect victim	Colour sensor		
Pick up victim	Slide victim on conveyor belt	Grab by the top of victim	Grab by the side of victim
Hold on to victim while exiting	Hold victim in front of robot	Hold victim on top of robot	Hold victim on the side of robot
Gear orientation - motor : wheel	Small : Large = Increasing torque	Large : Small = Increasing speed	1 : 1 = Same speed as motor
Size	Small	Medium	Large
Position of ultrasonic sensor	In front of robot at the top	In front of robot at the bottom	In front of robot on the side
Position of colour sensor	In front of robot in the middle	In front and on top of robot facing down at an angle	In front of robot on the side
Position of touch sensor(s)	In front of robot at the bottom	In front of robot in the middle	In front of robot on the side

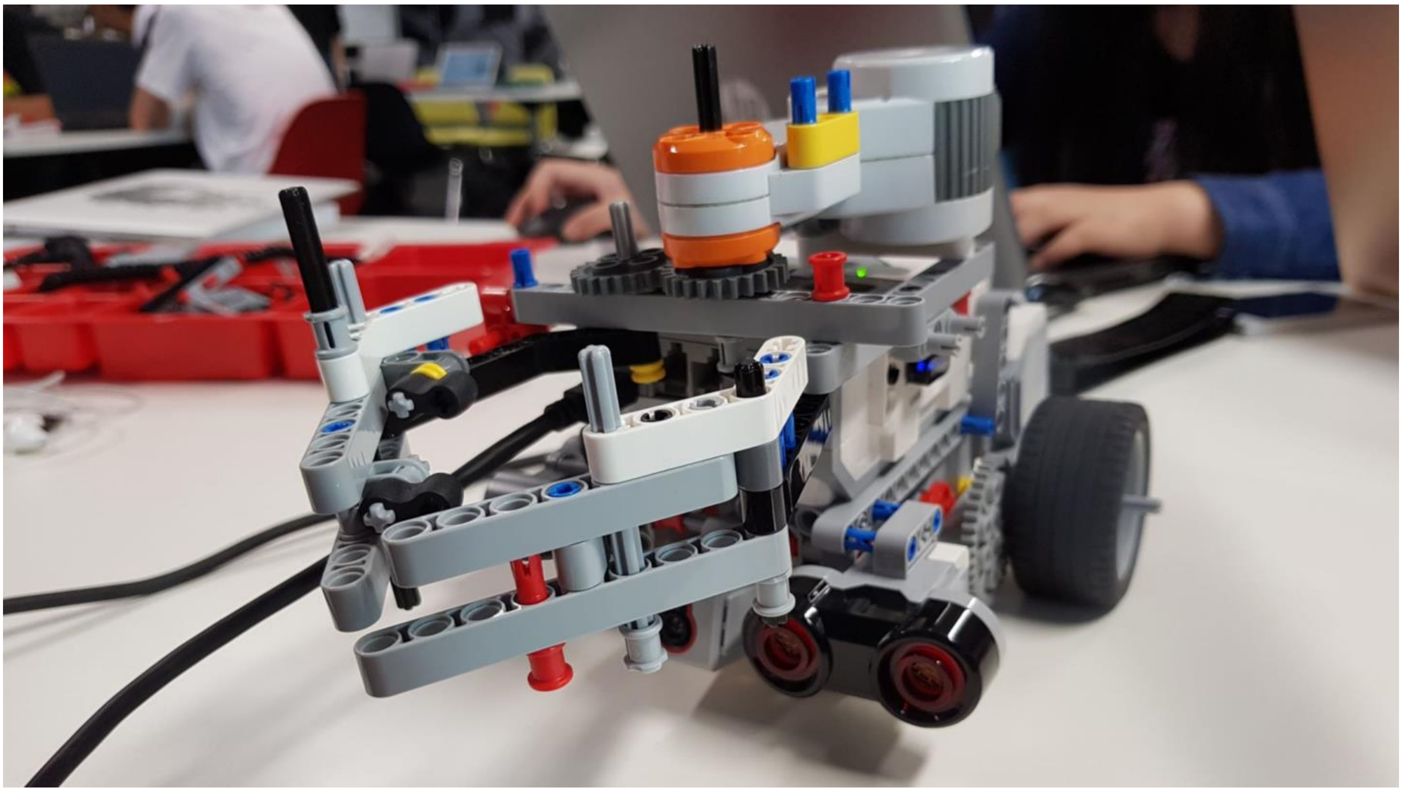
7.2 Appendix B: Rules and Restrictions

Rules	Restrictions
Navigate a maze	Time limit of 5 minutes
Finding the victim	Cannot push any wall away from its original place for more than 3cm
Signal with a sound or flashing light when victim is detected	
Pick up the victim with a grabbing mechanism	
Transport victim back to start of maze	

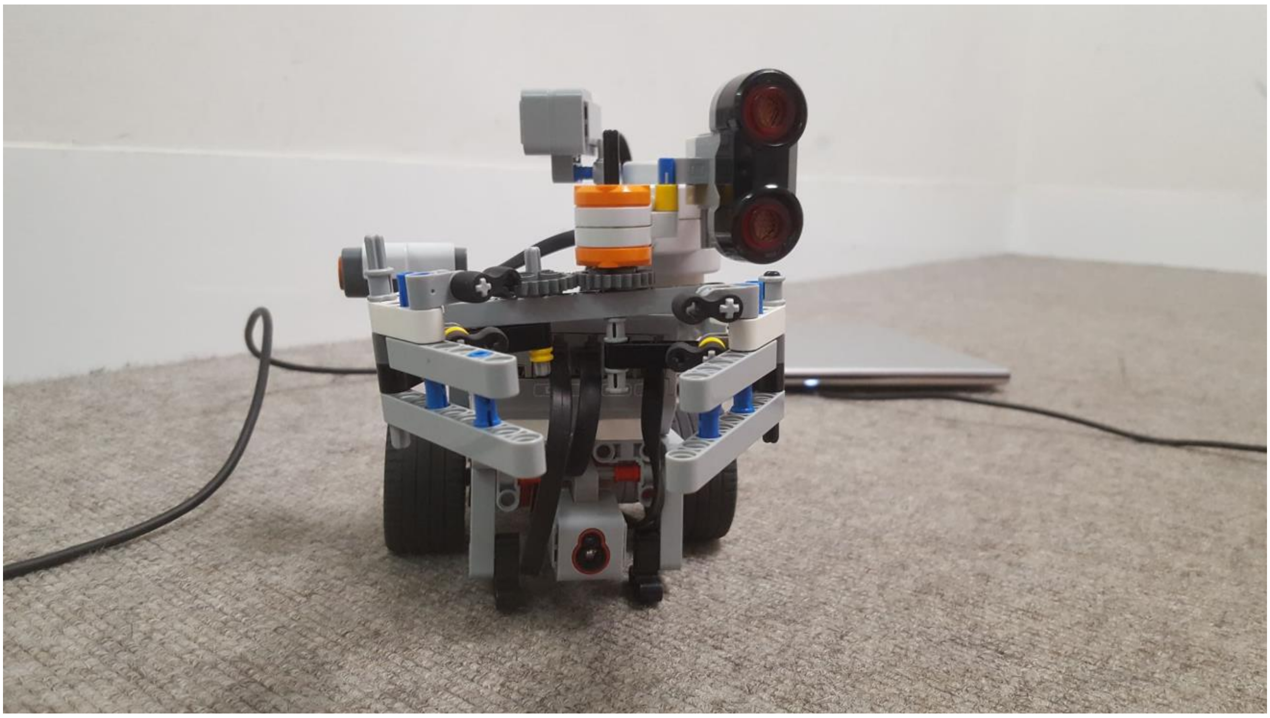
7. 3 Appendix C: Photos of Design Prototype #2 Before Finalised Design

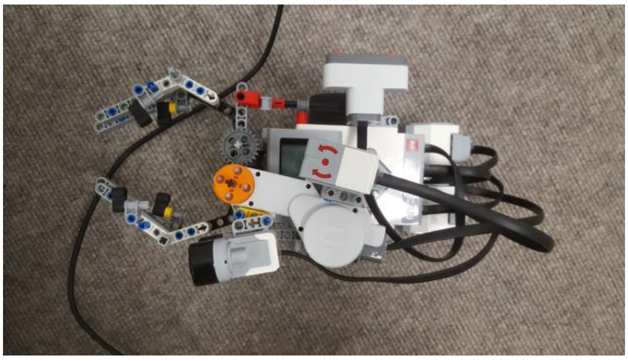
7.3.1 First Design

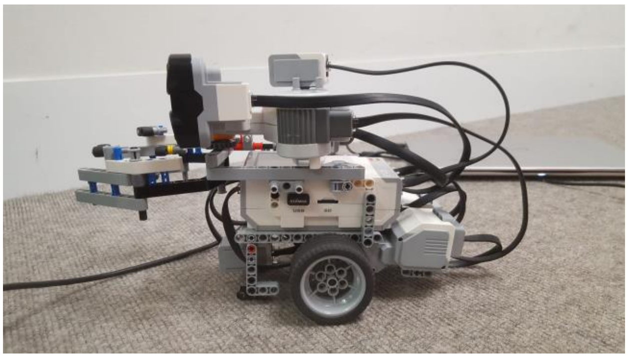


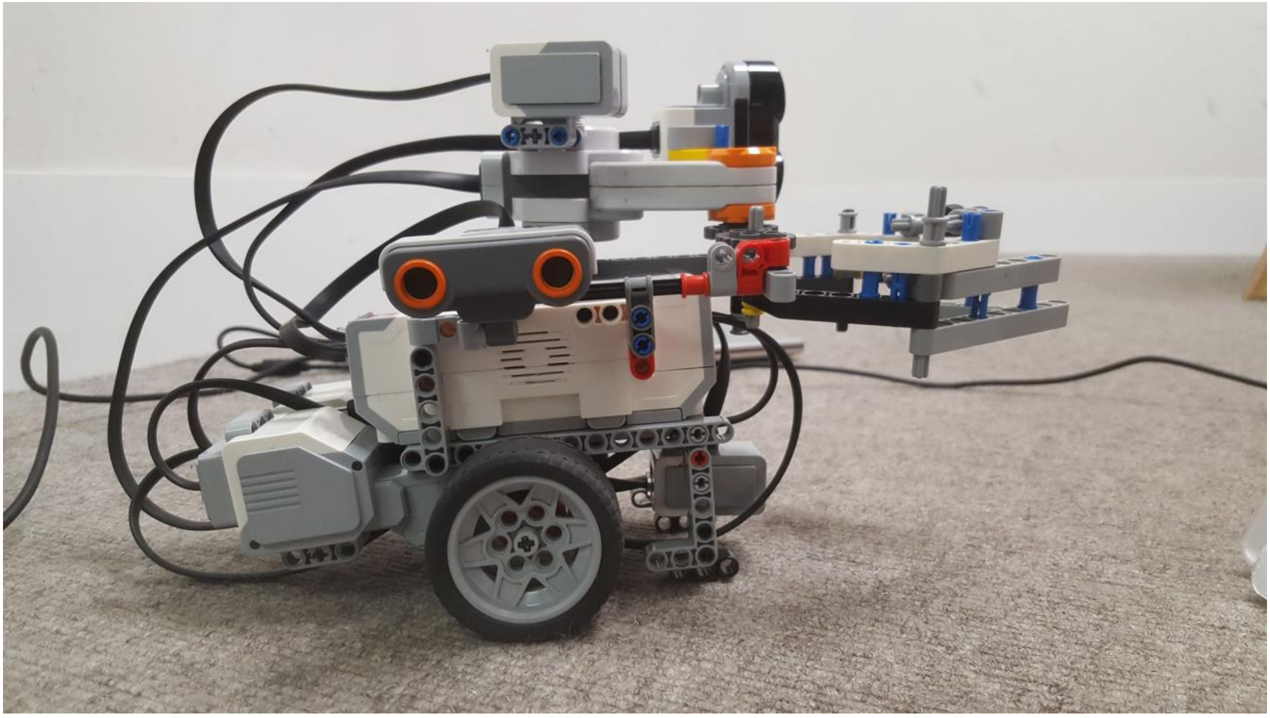


7.3.2 Second Design

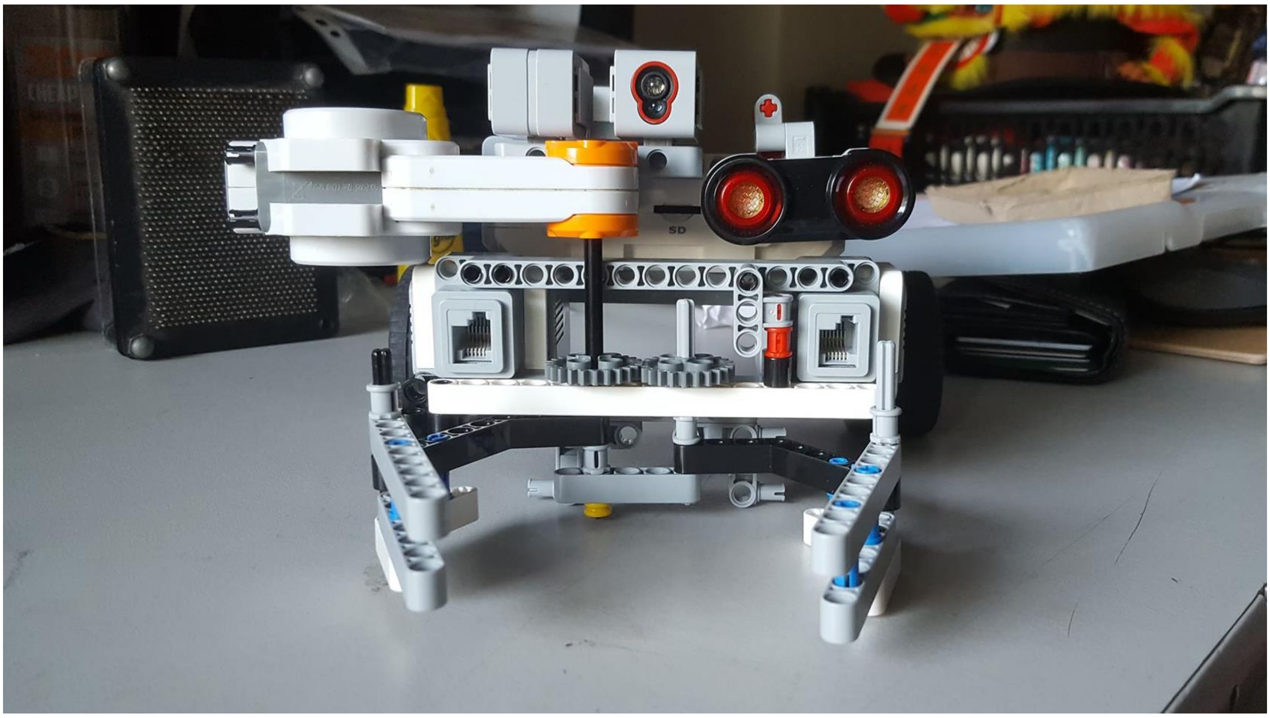


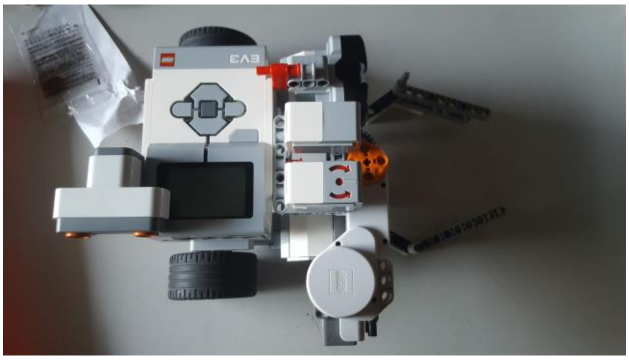


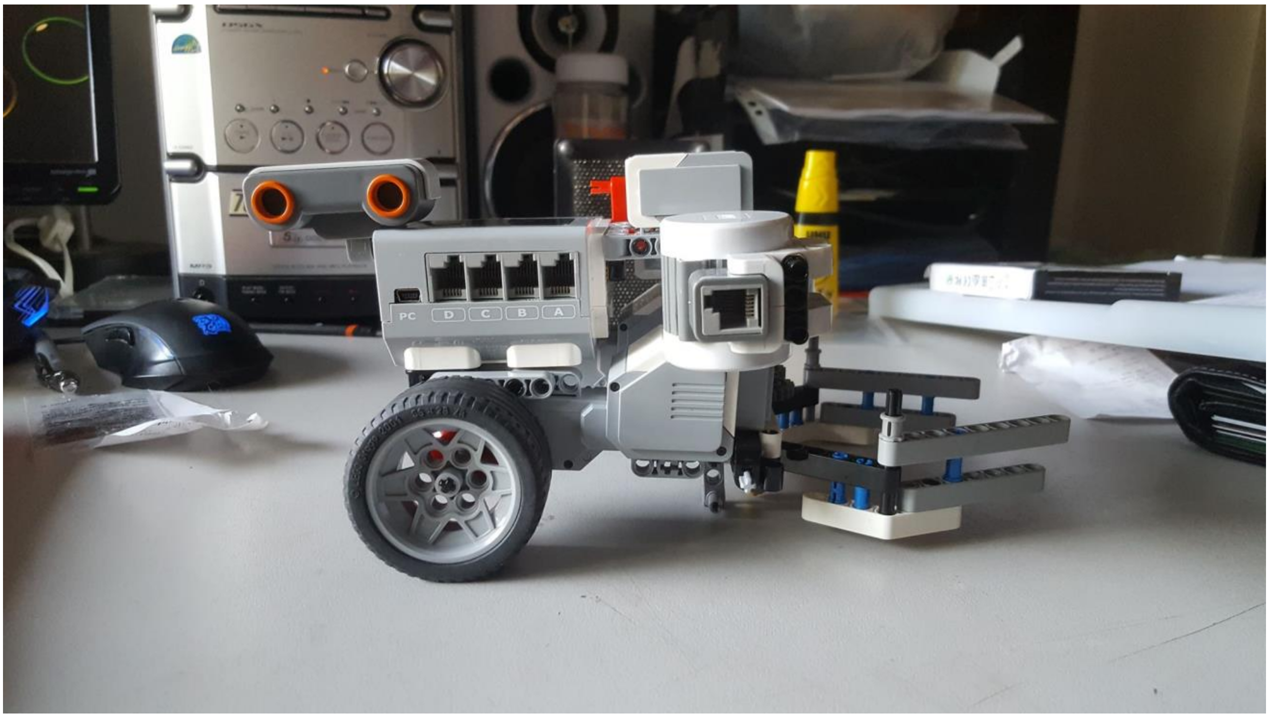




7. 4 Appendix D: Photos of Design Prototype #3 Before Finalised Design







7.5 Appendix E: Results of Final Testing

Maze	Go to victim (/4)	Find victim (/2)	Pick up victim (/2)	Return to start (/4)	Total (/12)
1	4	2	2	4	12
2	4	1	1	0	6
3	3	2	1	0	6