Rover Design Report

Group F04

Members:
Suowei Hu (u6966459)
Nur Syazwana Binti Mohd Nor Azman (u7218338)
Armando Jose Castelo (u7231943)
Zixuan Gao (u7306160)

Table of content

Executive Summary

This report delivers an analysis and evaluation of the prototyping and testing process of an autonomous maze solver, primarily referred to as a "Rover". Methods of analysis include measurements of rover dimensions, deviation of forward movement, distance, and angle tests of a sonar component as well as deviation in degrees of the Rover's turning motion. Other testing methods include evaluating the maximum and minimum ground speed value and optimal battery voltages needed for sufficient rover functions. These tests are conducted to ensure the best rover functionality by validating a list of design specifications. This list represents conditions to follow during prototyping and testing. Results from the analysed data provides validation towards the design specification values produce a rover favourable for autonomous maze solving. Values outside marginal range will produce multiple errors during rover movement.

This report finds the performance of the final prototype to be mainly positive. However, there were limitations that require further regulation: sensitivity of electronic components, measurement tools and integrity of chassis during movement. Recommendations were discussed which include the usage of uniformed and strengthened chassis parts as well as creating a more adaptable code algorithm.

Introduction

In the current modern era, automated robot has become more and more populated in the world, we see it working in multidisciplinary working environment around our daily lives. We see it at our dressing table, bank's front desk, factories. And in the past 5 years autonomous cars has been one of them carrying big headlines among the tech giants, despite the ethical issue, it is also worth our attention how it functions as a vehicle and avoid the obstacle. In the rover project in Discovery Engineering, we try to understand how it is solved as an engineering project with a grain of salt, via a relatively ground-up entry level project of automated rover built around micro-controller. In this report we will present to you how we defined problem scope and its requirements; Our designs; As well as how we test it, validating its performance against the requirements; And wrapping it up with some of our recommendation towards plausible future improvements.

Project Requirement

Project Scope

Engineering project is all about solving problem, archiving a specified value with limited resources and within some timeframe. In this rover project of Discovering Engineering course, we are asked to make a design of an automated rover capable of navigating through a maze using a sonar module to detect walls and avoid collisions. The rover follows programming from the onboard Arduino, receiving input from the sonar module, HC-SR04, to then decide whether the rover can move forward unobstructed or detect any walls that would prevent it from moving any further. The responses coded into the rover should allow it to complete a given maze without touching the walls with any part of the chassis.

As part of the design specification, we were restricted to use the provided components in the construction of the rover. These components include:

- An Arduino Uno Board
- Motor related: the SN754410 H-Bridge, 2 Micro Metal Gearmotors, 2 plastic wheels, a Ball Caster with a 1" plastic ball,
- Sensor related: the HC-SR04 sonar module, an 9G Micro Servo Motor,
- Other: batteries and wires to power the rover, a small breadboard,

With these components, we produced a rover with two engine-attached wheels that could be manipulated separately, along with a third wheel to hold up and balance the chassis and on-board components of the rover. With the provided components, we were able to create a list of design specifications the rover would need to fulfil and ensure completion of the given task within the given budget and time frame of the project.

Design Specification

Costs: Since there was no specific chassis provided, it should be noted that this had to be procured separately from the given components, in addition of recyclable materials. Additional expenses were made limited, so the values presented reflects the cost of items purchased for the creation of the rover.

Dimension: The given maze was composed of 19 by 19-centimetre squares, so our rover would need to fit within that space and turn without hitting any walls, hence we limit its chassis dimensions.

Weight Carriage: While navigate the maze, the rover must maintain its shape and structure while moving, so the chassis should be able to carry at least a weight of all the components combined. A time limit was also set for the rover to complete the maze, this means a rover of higher ground speed is always welcomed.

Motor Related: Meanwhile, our rover is controlled over the code on Arduino board, how it reacts following the "instructions" given by code are one of the keys to the project success. Hence, we measure motor's performance of accuracy on straight line motion and turning to capture this.

Sensor Related: The sonar sensor is the only "point-of-contact" our rover has to the environment, rover determine distance of wall through it, hence we also need it to be as accurate the precise as possible to ensure the right action is carried, and false positive do not happen.

A full list of these design specification is listed below (an enlarged version will also be attached in the appendix):

	Design Specification	Note	Quality	Units	Ideal Value	Marginal Value
1	Price	(exclusive of replacement for broken components)	Australian Dollars	$ (AUD)	$0	≤$20
2	Dimension	(by measuring spacing between outer most components)	Length, Width	Centimeter	≈ 10 x 10	≤ 14 * 14
3	Overall Weight	(Inclusive of the batteries, and any other external attachments)	Mass	Kilograms	≈ weight of components	2 x weight of components
4	Hold component in palce	(even during motion)	Carrying Capacity	Grams	2 * weight of rover	weight of rover
5	Sensor Performance Resolution	(performance under varies distance)	Maximum resolution the rover can measure	Centimeters	± 1 cm	± 5 cm
6	Sensor Performance Varies Angles	(test for the performance under varies angles)	Maximum angle the rover is able to measure correctly	Degrees	30°	10°
7	Sensor Performance Varies Distance	(test for the performance under varies distances)	Max/min distance the rover is able to measure correctly	Centimeters	0.0cm ~ infinity	5.0cm ~ 10.0cm
8	Motor Performance: Ground Speed	N/A	Distance over Time	Meters per second	456 cm / 90 second	456 cm / 180 second
9	Motor Control Forward Motion	N/A	Deviation in movement	Centimetres	± 0 cm	± 2 cm
10	Motor Control Turning	(measuring the amount of error when making a 90° turn)	Deviation in turning angle	Degree	± 0 degrees	± 5 degrees

(Rover Project - Design Specifications)

Rover Design

Overview

The goal of this engineering project is "a rover that is capable navigating through a maze using a sonar module to autonomously detect walls and avoid collisions". Such gigantic question could be broken down into three parts: Movement, Obstacle Detection, Route Planning / Logic (i.e. Code). We brain stormed and came up with four design concepts, each having different strength and weaknesses, and we chose one among the four as our final decision to implement (A complete list on details of those designs can be found in the appendix.)

Design Details

The design we end up with is the "Layered Burger". Assessing it against our design specification, it is advantages in having no upfront costs ("Tank" need extra rubber track), and the chassis material is very accessible ("Siege Towel" need solid cardboard like phone packaging);

Basic Chassis: The motors, ball casters, and batteries are fixed at the bottom layer, to keeping a low centre-gravity. Notice that this layer carries the weight of most component in the build, hence is build using more rigid cardboard, this reflects to the specification of weight carriage. The H-Bridge are wired properly to enable to Arduino to control motors in both speed and direction, without the need of rewiring. Moreover, the "1,2,3,4EN" pin on the H-Bridge are wired to a digital output pin that's capable of outputting different voltages via PWM, this helps resolving motor's un-synchronous by fine tuning the voltages, allowing the rover moving a more accuracy and predicted manner (specification6,7).

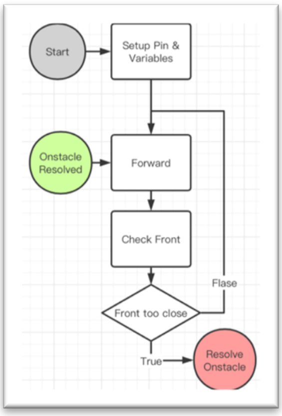
Movement: The sonar (ultrasonic) sensor sits on top of the servo motor, and together with the Arduino and breadboard, they sit on the top layer. The reason behind having the sensor's position is that it is only capable of making reliable read when the clamping angle (between obstacle's surface and connection line from sensor to the object) is within a range ±15°. In the maze we encounter walls of ±45°, we can certainly have sonar sensor fixed point to one direction and move the rover body to make it align with the wall, but thus increase the likelihood of hitting the wall. Hence the only remaining solution is to have it sit on a rotatable object like servo. In fact, this approach also raises the sensor up, making it less likely to get interference from the wave reflecting from the ground, making it even more accurate and precise.

Programmed Logics: For a rover to successfully navigate through the maze, its programmed code can be divided into 4 phases:

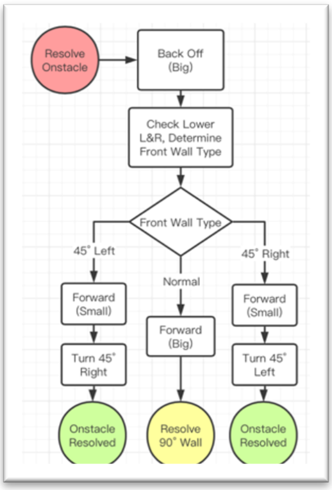
1. Detecting the wall and stop
2. Detecting front wall type
3. (Optional) Detecting left/right wall difference
4. Making movement

An enlarged process map showing these logical steps can be found in appendix. And The Arduino INO code file itself is also attached.

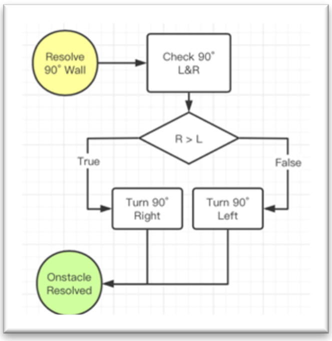
Phase 1: The rover keeps moving forwards and keeps taking the sonar measurement at the 0° centre direction, until a close enough obstacle is detected, then proceed to phase 2. As one of the results from sonar assignment, that is: taking the median value from a normal distributed dataset, a more accurate and precise measure can be determined. Here, each measure is taken by using the median of five measurement, by doing this we trade off milliseconds level difference in respond time for a much higher accuracy and precision. This corresponds to the design specification 4,5 of "Sonar Performance".



Phase 2: Once a close enough wall is detected, the rover will detect the type of wall. In the assignment scenario given, we are facing 3 types of walls: wall perpendicular to rover's moving path, wall that's 45° clockwise and anticlockwise to the path. One difficulty we face is for the last two type of wall, the rover's measure to it is always higher than actual; in another word, when entering from phase 1, the rover is actually already near touching the wall (≤2.0cm); and we know the sonar won't get accurate measure under 2.0cm distance. Consequently, we program it to back off a little and turn the sonar sensors aligning it with the wall in order for more accurate measure.



Phase 3: After we know the wall type, we proceed to the next step. If normal wall then returns to the position where it backed-off from, measure distance to the closest left/right wall, and turn towards facing the one of longer distance, then reset to Phase 1. Otherwise, if 45° wall, play a programmed motion of turning 45° right or left, then reset to Phase 1.



These logics/codes combine allowing the rover able to detect walls, move accordingly to the wall type, and navigate with proper route planning; Together enabling it to solve an unknown maze under the assignment scenario. (Again, an enlarged version of these low chart can be found in the appendix)

Prototype, Testing & Validation

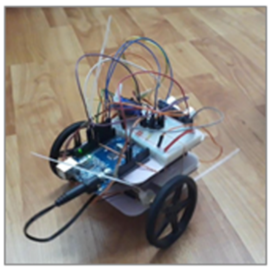
Overview

We executed the Rover build and testing in conjunction with each design specification. We used recycled materials in the body of each prototype. We prioritised vertical dimension over horizontal to ensure it can navigate within the dimensions of the maze. The chassis build was a critical aspect within the design and underwent iterations to safeguard its integrity throughout each testing for optimal consistency and reduce uncertainties in results. Individual specifications were tested differently.

Prototype

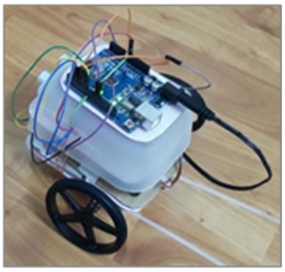
We mainly have three stages of prototypes. Each enhances based on the bits of knowledge and lessons we learned from the previous, compared to their ancestors. They advantage various ways: dimension, motor performance, motor accuracy, sonar performance.

First Prototype: In our very first prototype, we built around on the "Layered Burger" design, and the result is a rover made of cardboard that can perform all actions with some degree of errors. However, after some testing is done, the bottom cardboard starting to bend concave downwards, resulting rover's unexpected behaviors. For instance, motors stop unexpectedly.



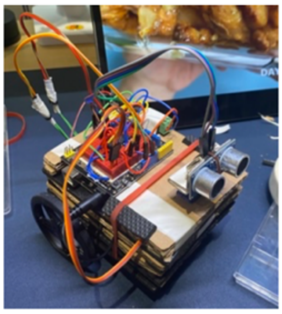
First Prototype

Second Prototype: To solve the issue in the first prototype, we used a more rigid material for the bottom layer (two cardboard stacked as one). And further, shrink the form factor via a plastic box (15x15cm to 14x14cm). The result is very gratifying, the unexpected motions are primarily decreased, and the rover was less likely to collide with walls. But due to the motors' un-synchronous issue, it is still space for improvement.



Second Prototype

Final Prototype: Through the lessons we learned from first two prototype, in this iteration, we focus on making the rover as rigid and solid as possible, such that all possible errors occurring are systematic and can be solved via fine-tuned code; And We used a cardboard + foam new constructing method to make this happen. An instance of systematic error would be a consistent clockwise tiering of 1° while moving forward, for solving by giving a slightly lower voltage output on the left motor via feeding lower voltage on the 1,2EN pin.

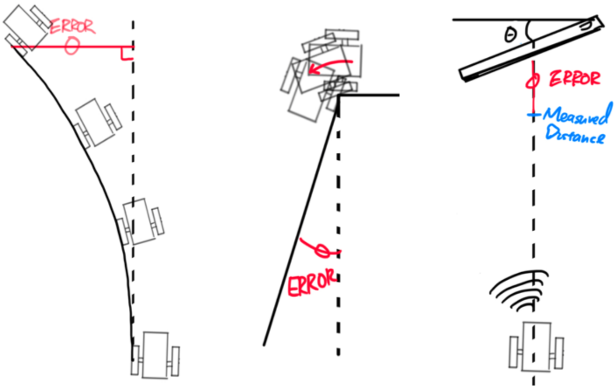


Last Prototype

Testing Results

Motor: To measure the deviation in forwarding movement, we first placed the rover on a paper and marked on one wheel's starting point, turn it on and let it travels for 30cm, then connect the position of a wheel to its starting point. The perpendicular distance from such line to an ideal travel path is taken as the deviation/error of this go. Moreover, Time taken to travel such 30cm is also recorded for testing of ground speed; error turning angle is also measured similarly. Finally, we mark the wheel's position before and after the 90° turn, then using a protractor to measure its angle different to "true" 90°.

Sensor: To measure the sonar's performance under varies distances and angles, we unplug the power of its servo/motor; Use 12.9-inch iPad stimulating walls, place it with various distances and angle in front of the sensor, note down the actual distance/angle of this obstacle, and record rover sensor's measure for multiple times.



(Testing Experiment Illustration)

The testing results in are attached in appendix as table, they comes from our final prototype. To eliminate any uncertainties in a single measurement, we take 10 tests for each of the testing; The median, std, or other statistical measures are calculated for later verification works.

Validation

In testing results, we calculate statistical measures of "the absolute value of errors." From the statistics, we could see the mean value of all the errors are under the marginal value with:

- Mean forward movement error of 0.5cm per 30.0cm
  and is under the requirement of 2.0cm.
- Mean turning error of 1.0 degree per 90.0 degrees
  and is under the requirement of 10.0 degree.
- Mean distance/bearing range within the acceptance range of design specification.

However, they are still far from the ideal values because it is very different when compared the ideal condition. By comparing the real world with ThinkerCAD, we will show you a bit more what we mean by this via two key points we've found.

Firstly, in simulation, the speed of the motor's rotation is precisely proportional to the voltages given, such that increasing the voltage increases the speed linearly. However, with the motor in the physical world, it only moves when the voltages provided were higher than the threshold of 1.95V, suggesting an internal resistance in the motors. Only when the voltage supply is enough can a torque be produced to overcome this resistance and move the motors. It is also worth noting that in perfect condition, there is no inter-connection and wire resistance.

Realistically, loose connection or resistance of copper wires (0.0171 Ohm · mm^2/m) can eventually result in ultimately lower voltage output to the motor, reducing the speed.

Secondly, in simulation, the sonar sensor has relatively better 0accuracy and precision. For example, if you simulate an object being 10.0 cm in front of the sonar sensor and take multiple reads, the results fall precisely around 10.0cm. In actual conditions, our sensor suffers from environmental conditions such as the detection object's surface not being perpendicular to the wave ejecting unit, or there are other obstacles near reflecting waves. These variables are independent of the measuring and bring uncertainties. The results are mostly values nearing the true values. However, erroneous readings can appear randomly. In the rover code design, errors were minimised by taking the median of 5 readings as a single read. Adjustments in the code also allowed us to attempt any sonar recordings to be within values we were sure would provide reliable readings, such as reading within 20 centimetres or having the sonar face any angled walls directly to reduce such errors.

Future Recommendation

In developing the prototype and design for this rover, several issues arose from using different kits and chassis materials. And this includes the different sources of error from using different motor and sonar sets and the recycled materials used in creating the chassis not being able to sustain its integrity throughout the production period. If possible, using the components of higher accuracy would be preferred, for instance, replacing brush motors with stepper motors able to control the exact RPM.

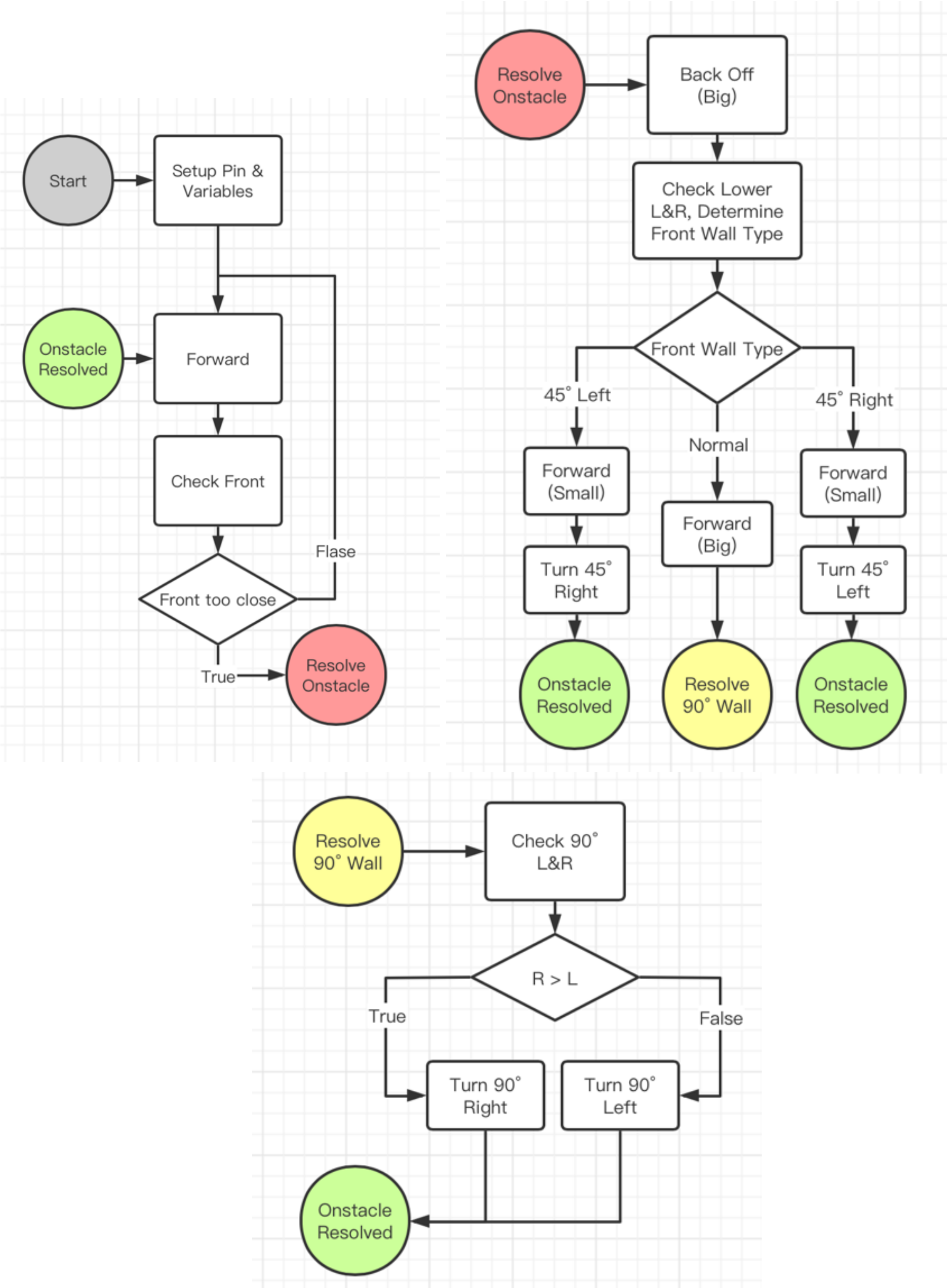
The variation in rover also built results in uncertainties. When using the same code but a different rover, the actual behaviour might not be 100% matched with the programmed behaviour. While using multiple rovers allows the team to work separately in solving different issues, working on a single rover could help with a more fluent and consistent experience with rover's designing, programming, and testing.

A more direct guideline for testing and prototyping might also be helpful for teams to proceed on the same page when working concurrently as we did for this project. This was particularly important in the coding, which was resolved by using meticulous notes to indicate the separate variables each member may have set up when programming their rover algorithm.

Conclusion

As a result of the rover project, we have followed through the engineering process of defining problem; developing requirements; brainstorming for concept; and design, implementing prototype; testing, and validation; and lastly reflection. We have not only successfully made an automated rover that solves the project's ultimate goal of solving the maze, but also explored the varies aspects in such project that could introduce uncertainties and risks, and most importantly learnt the mechanism behind these processes through practice. Consequently, such valuable experience could later be integrated to our future career and making us more competitive in the labor market.

Appendix



(Rover Project - Programming Logics)

	Design Specification	Note	Quality	Units	Ideal Value	Marginal Value
1	Price	(exclusive of replacement for broken components)	Australian Dollars	$ (AUD)	$0	≤$20
2	Dimension	(by measuring spacing between outer most components)	Length, Width	Centimeter	≈ 10 x 10	≤ 14 * 14
3	Overall Weight	(Inclusive of the batteries, and any other external attachments)	Mass	Kilograms	≈ weight of components	2 x weight of components
4	Hold component in palce	(even during motion)	Carrying Capacity	Grams	2 * weight of rover	weight of rover
5	Sensor Performance Resolution	(performance under varies distance)	Maximum resolution the rover can measure	Centimeters	± 1 cm	± 5 cm
6	Sensor Performance Varies Angles	(test for the performance under varies angles)	Maximum angle the rover is able to measure correctly	Degrees	30°	10°
7	Sensor Performance Varies Distance	(test for the performance under varies distances)	Max/min distance the rover is able to measure correctly	Centimeters	0.0cm ~ infinity	5.0cm ~ 10.0cm
8	Motor Performance: Ground Speed	N/A	Distance over Time	Meters per second	456 cm / 90 second	456 cm / 180 second
9	Motor Control Forward Motion	N/A	Deviation in movement	Centimetres	± 0 cm	± 2 cm
10	Motor Control Turning	(measuring the amount of error when making a 90° turn)	Deviation in turning angle	Degree	± 0 degrees	± 5 degrees

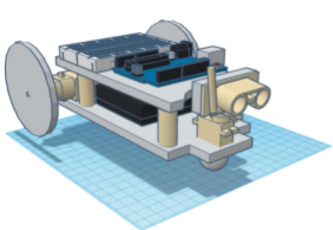
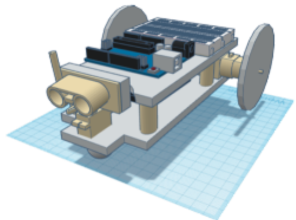
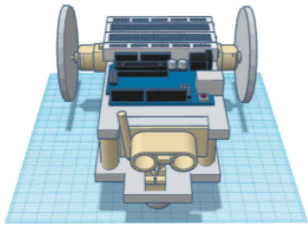
(Rover Project - Design Specifications)

```
void loop() {
    // while(true){make_leftTurn(); delay(1000);}
    // while(true){make_rightTurn(); delay(1000);}
    // while(true){myservo.write(SM_MID_L-23); double dist = read_dist(); println(String(dist));}

    forward();
    double dist = read_dist();
    double dist_l = 0;
    double dist_r = 0;
    double dist_c = 0;
    // println(String(dist));
    if(dist < 7.5){
        stop();
        delay(500);

        int wallType = check_frontWallType();

        stop();
        delay(500);


        switch (wallType){
            case NORMAL_WALL:
                // go back to the original position
                forward();
                delay(BACK_OFF+200);
                stop();

                myservo.write(SM_LEFT);delay(500);
                dist_l = read_dist();
                myservo.write(SM_RIGHT);delay(500);
                dist_r = read_dist();
                myservo.write(SM_CENTRE);delay(500);
                dist_c = read_dist();

                println(String(dist_l));
                println(String(dist_c));
                println(String(dist_r));
                println("");

                if(dist_c >= dist_l && dist_c >= dist_r){stop();}
                if(dist_r>dist_l){make_rightTurn();}
                else if(dist_l>dist_r){make_leftTurn(); }
                else{make_rightTurn();}
                break;

            case FORTYFIVE_WALL_ON_LEFT:
                // go back to the original position
                forward();
                delay(BACK_OFF-300);
                stop();

                // while(true){println("FORTYFIVE_WALL_ON_LEFT");stop();}
                make_fortyFive_rightTurn();
                break;

            case FORTYFIVE_WALL_ON_RIGHT:
                // go back to the original position
                forward();
                delay(BACK_OFF-300);
                stop();

                // while(true){println("FORTYFIVE_WALL_ON_RIGHT");stop();}
                make_fortyFive_leftTurn();
                break;
        }

        delay(2000);
    }
}
```

(Main logical code for solving the maze,
for the complete code you could find it at: https://anu365-my.sharepoint.com/:f:/g/personal/u6966459_anu_edu_au/EmFMRaGq_41HvpMlbjiiA3gBfG5pAwt-QzmuzzIeTm7MYw?e=l1KvL7)

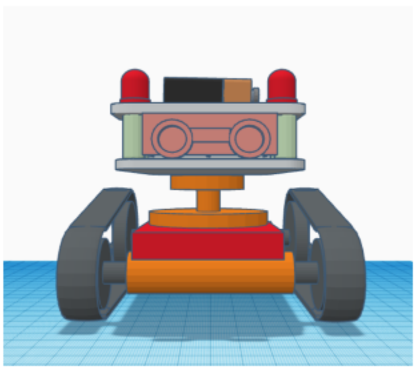
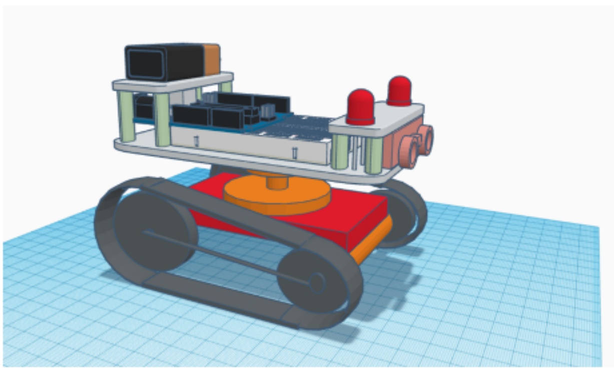
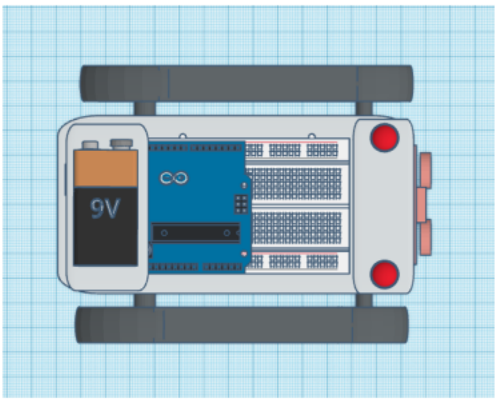
Rounded burger (Suowei Hu)

- Very compact, easy to get components and make.
- ≥2 layer of space (1 for battery pack, 1 for breadboard).
- If needed, add another layer of PCB plate can hide the wires.

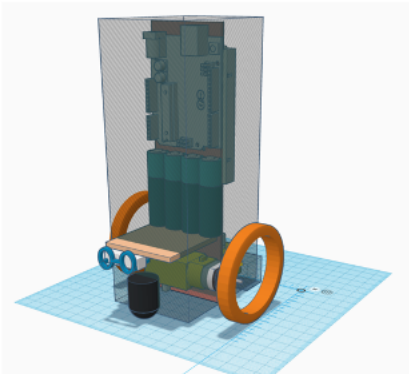
"Tank track" (Wana Azman)

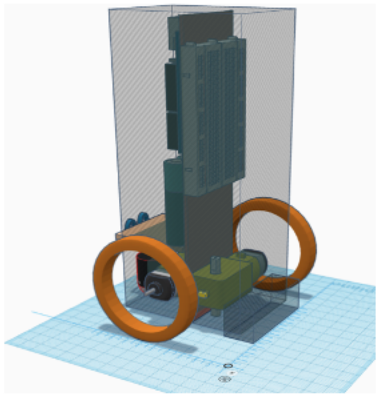
- Require extra parts (3wheels +1track pair)
- High stability
- High Precision movement
- Accurate Rotational Movements
- Large Mass

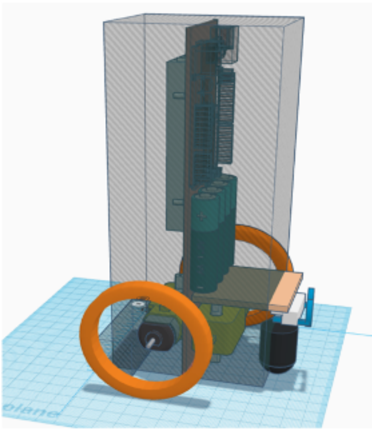




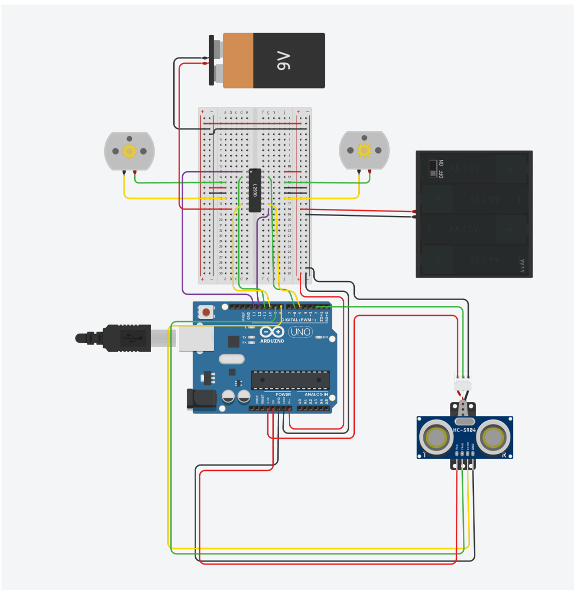
Siege Tower (Ar-J)

- Can fully turn in place with minimal displacement.
- Smallest area possible for rover dimensions.
- Needs strong material to be able to hold components aloft.
- Off – centre motors may require attention in programming algorithm or in construction.
- Higher centre of gravity may affect balance of the rover.









Trial	Error of left wheel (cm)	Error of right wheel (cm)
1	0.2	0.3
2	1.0	1.0
3	0.2	0.1
4	0.9	0.9
5	0.4	0.5
6	1.0	1.0
7	0.4	0.4
8	1.0	0.9
9	0.4	0.3
10	0.1	0.2
11	0.1	0.2
12	1.0	0.9
13	0.9	1.0
14	0.9	1.1
15	0.5	0.5
16	0.6	0.5
17	0.7	0.5
18	0.4	0.4
19	0.4	0.6
20	0.3	0.4
Mean	0.6	0.6
Median	0.5	0.5
Std Dev	0.3	0.3

(Absolute Error of rover moving in straight line after travelling 30 centimetres)

Trial	Error Right Turn (°)	Error Left Turn (°)
1	0.0	0.0
2	1.0	0.0
3	1.0	1.0
4	2.0	0.0
5	0.0	2.0
6	2.0	1.0
7	1.0	0.0
8	2.0	1.0
9	0.0	1.0
10	1.0	0.0
11	3.0	1.0
12	3.0	2.0
13	4.0	0.0
14	0.0	1.0
15	1.0	0.0
16	2.0	2.0
17	1.0	1.0
18	0.0	0.0
19	1.0	1.0
20	3.0	1.0
Mean	1.4	0.8
Median	1.0	1.0
Std Dev	1.2	0.7

(Absolute Error of Rover Turning Degrees when Turning for 90°)

Trial	Error at centre position (cm)	Error at 45° Clockwise (cm)	Error at 45° Anti-clockwise (cm)
1	0.0	18.0	39.0
2	0.1	18.3	38.9
3	0.1	18.0	38.9
4	0.1	18.0	40.0
5	0.1	18.6	38.7
6	0.1	18.0	38.9
7	0.1	18.0	38.9
8	0.1	18.0	38.0
9	0.1	18.0	40.0
10	0.0	18.6	38.0
11	1.0	18.0	38.0
12	0.1	18.0	39.0
13	0.1	18.6	40.0
14	0.1	1177.0	44.0
15	0.0	1176.0	40.0
16	0.1	1177.0	40.6
17	0.1	1177.0	40.0
18	0.1	24.0	39.5
19	0.0	18.0	52.9
20	0.1	18.6	39.0
Mean	0.1	250.2	40.1
Median	0.1	18.2	39.0
Std Dev	0.2	475.3	3.3

(Absolute Error of sonar sensor when wall is 10.0cm from rover at different bearing)

Trial	Error when wall at 5.0cm	Error when wall at 10.0cm	Error when wall at 20.0cm
1	0.3	0.2	0.2
2	0.4	0.0	0.0
3	0.2	0.3	0.4
4	0.3	0.2	0.5
5	0.4	0.7	0.6
6	0.3	0.2	0.6
7	0.5	0.3	0.5
8	0.4	0.3	0.5
9	0.3	0.3	0.3
10	0.3	0.3	0.7
11	0.3	0.3	0.7
12	0.2	0.3	0.9
13	0.3	0.2	0.7
14	0.3	0.2	0.8
15	0.3	0.0	0.8
16	0.3	0.1	0.8
17	0.3	0.1	0.9
18	0.2	0.0	1.1
19	0.3	0.1	0.8
20	0.3	0.3	0.7
Mean	0.3	0.2	0.6
Median	0.3	0.2	0.7
Std Dev	0.1	0.2	0.3

(Absolute Error of sonar sensor when wall is perpendicular to rover at different distance)