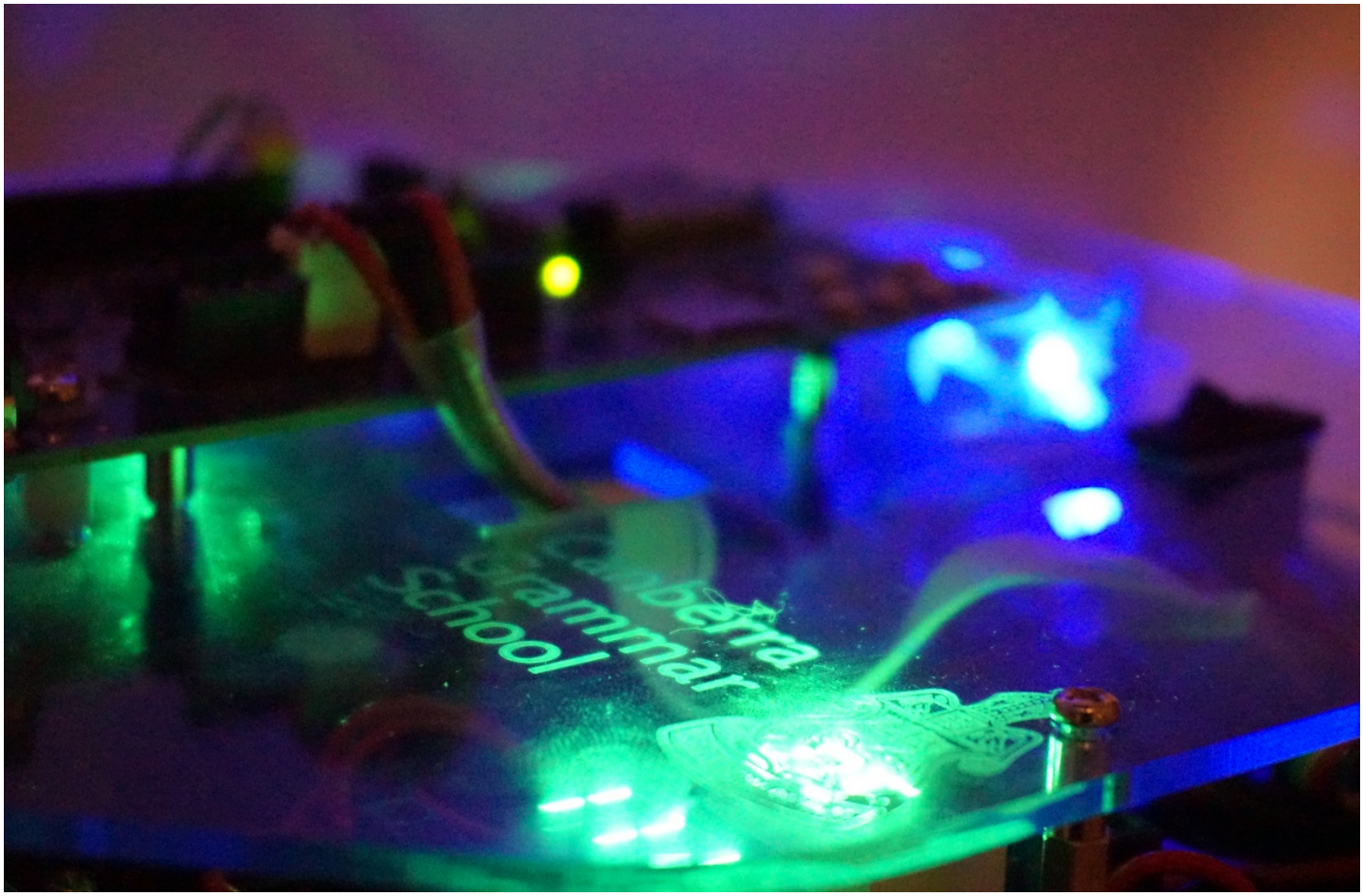
2016 RoboCup Internationals

Australian Rescue Maze Team



Project Proposal

Prepared for: RoboCup Junior Rescue B

Prepared by: Rory Wade, Joseph Fergusson and Ines Kusen

20 May 2016- 23 July 2016

EXECUTIVE SUMMARY

Objective

This journal shows the design process used to create our robot for the RoboCup Rescue B competition in Leipzig, Germany. It outlines the challenges that we encountered throughout the lead up to the competition and how they were resolved. This journal will guide you through our adventure for RoboCup Rescue B.

Goals

Our main goal throughout this journey was to learn a lot, but also to have fun along the way. Being a part of RoboCup Junior Internationals is not something that everyone can do and it is an outstanding opportunity where we can learn from and interact with a worldwide community of engineers, scientists and experts working in the field of robotics. We have participated twice before in this competition, and have improved our robot each year. This year, we hope to improve our results still further and ultimately become more competitive in future competitions of Rescue B.

Aim

To create a custom-built robot and program it through Intel Edison and Open CM 9.04 to compete at the Robocup Rescue B Competition, 2016, in Leipzig, Germany.

Project Outline

- Rp - Lidar sensors to navigate maze
- Heat sensors to sense heat spots
- LEDs to shine on heat spots
- Drop Mechanism to drop rescue kits
- Light sensor to find black tiles
- Ability to continue despite obstacles (e.g. up and down hill, toothpicks, tissues, etc.)
- Communication between the Intel Edison and the OpenCM
- Servo directional changer

BUDGET

Bill of Materials

Description	Quantity	Unit Price	Cost
Robot Peak Lidar	1	$ 600.00	$ 600
Heat Sensors	3	$ 15.00	$ 45
Dynamixal AX-18	4	$ 87.00	$ 348
Resistors	28	$ 0.20	$ 5.60
Wires (m)	3	$ 10.00	$ 30
Solder	1	$ 13.00	$ 13
Teensy	1	$ 15.00	$ 15
Screws/bolts/washes	89	$ 0.80	$ 71
Capacitors	3	$ 0.20	$ 0.60
Wire Clips	6	$ 1.00	$ 6.00
LEDs	8	$ 2.30	$ 18.40
Batteries	16	$ 2.00	$ 32
Omni Wheels	1	$ 60.00	$ 90
Acrylic	2	$ 12.00	$ 24
Bearings	16	$ 0.50	$ 8
ABS Filament (per spool)	1.0	$ 35.00	$ 35.00
Headers for Pins	62	$ 0.25	$ 15.50
Heat Shrink (m)	2	$ 3.00	$ 6
Batteries	2	$ 30	60
Velcro (cm2)	16	$ 2.00	$ 32
Touch switches	6	$ 1.00	$ 6.00
Axels	12	$ 4.00	$ 48
PCB	1	$ 12.00	$ 12
IMU	1	$ 30.00	$ 30
Plastic dip	1	$ 30.00	$ 30
Servo	2	$ 15.00	$ 30
Miscellaneous	1	$84.80	$ 84.80
Total			**$1400.00**

Date: 11th January, 2016

Phase of Design Process: Design Criteria

Agenda:

- Review rules and robot
- Plan components to buy
- Order components
- Create a table listing the things that worked on previous robots and what needs to be changed
- decide on mapping algorithm
- Build Lego prototype of robot

Progress/Status:

1. REVIEWING LAST YEAR'S ROBOT AND DECIDING ON CHANGES

The first step was to create a table listing the pros and cons of the robot we used last year and to determine which characteristics worked well. In this way, we were able to decide on the work that lay in store for us. We reflected on our experiences from last year and the knowledge we had gained to help us improve our robot.

2. PLANNING PHASE

We decided last year that the robot needed a much more powerful micro processor, so we ventured from Arduino into using three separate boards to control different elements of the robot. We will be using the Intel Edison (for mapping), OpenCM (for motor movements using Dynamixel motors) and a Teensy (for all other sensors necessary for maze navigation). We also decided that Ines would learn the SLAM (Simultaneous Localisation and Mapping Algorithm) so that we could apply it to mapping in the maze. For SLAM, we would have to buy a LIDAR sensor. We unfortunately did not achieve this outcome this year as we couldn't grasp the concept this year. Rory will design and build the robot and program the low level processor. While Joseph will program the Mapping Algorithm on the higher level processor,

We also brainstormed and found new inspiration for the robot design. We went back to using the Omni wheels front and back , as we deemed this would give the robot better manoeuvrability. We were finally able to purchase a range of parts, some identical to last year's robot, and some (Dynamixels and LIDAR

sensor) completely new. Once again, the robot will be manufactured using the 3D printer and the Laser Cutter.

3. *BUILDING LEGO PROTOTYPE*

Rory has built the first Lego prototype of the robot's moving system. It is a completely new system that was researched and will allow the robot to manoeuver over large and uneven obstacles.

Issues/Solutions:

HARDWARE:

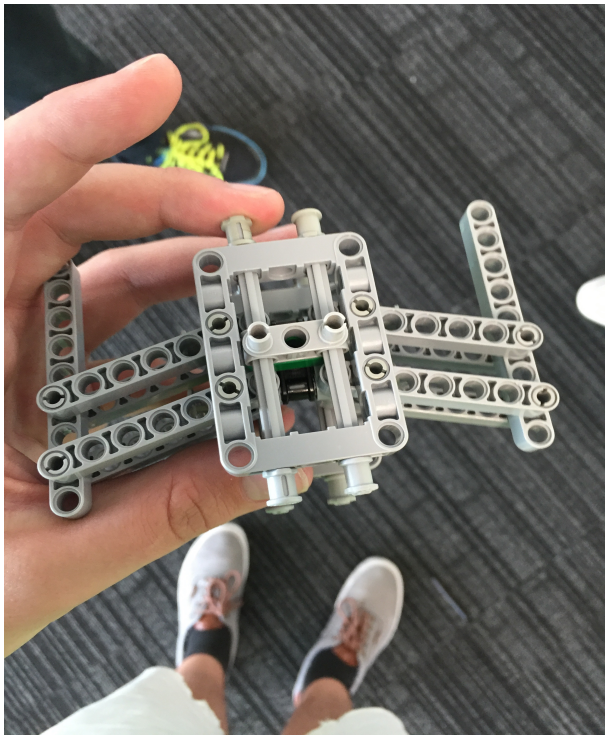
- As pieces are being shipped in from overseas it is unknown the time it will take to get to us. There is not much we can do until the parts get here! However, we can continue working with the parts we already have from last year's robot, such as the heat sensors.

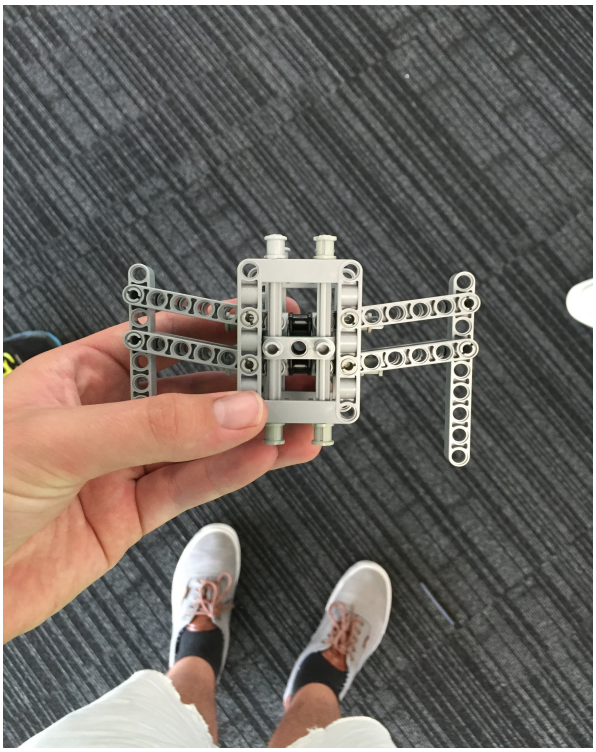
Software:

- We need to understand how to communicate between different boards using the I2C Bus and design a Slave Master program.

Solutions:

- Find other activities to achieve while waiting for the parts, such as re-testing the heat sensors and finding other component - Trial the Slave-Master program using two Arduino boards and LED's to understand how it works.





Date: 19th April, 2016

Phase of Design Process: Design Criteria

Agenda:

- Review rules
- Plan components to buy
- Order components
- Create a table listing the things that worked on previous robots and what needs to be changed
- decide on mapping algorithm
- start brainstorming robot design

Issues/Solutions:

HARDWARE:

- As pieces are being shipped in from overseas it is unknown the time it will take to get to us. There is not much we can do until the parts get here! However, we can continue working with the parts we already have from last year's robot, such as the heat sensors.

SOFTWARE:

- We need to understand how to communicate between different boards using the I2C Bus and the Slave-Master program.

Solutions:

- Find other activities to achieve while waiting for the parts, such as re-testing the heat sensors and finding other components

- Trial the Slave-Master program using two Arduino boards and LED's to understand how it works.

Date: 19th April, 2016

Phase of Design Process: Prototyping

Agenda:

- Build and design first Lego prototype

Progress/Status:

1. BUILDING FIRST PROTOTYPE

The first Lego prototype was made over many days with multiple iterations of the body mechanism. The prototype used the Lego technic pieces and utilised gears to make a functioning suspension. This allows us to find solutions to issues that might have arisen later in the build and also allowed us to get a good feel for the proportions of the robot.

Issues/Solutions:

HARDWARE:

- Stability

-Reliability

SOLUTIONS:

- Design the next version is an acrylic material which will be designed in illustrator and laser cut.

Date: 10th March, 2016

Phase of Design Process: Prototyping

Agenda:

- Build and design first Acrylic prototype

Progress/Status:

1. BUILDING FIRST PROTOTYPE

The first prototype was designed and built over the course of two weeks. It was laser cut on blue acrylic and the mechanism was then built and tested. It was challenging to find the right ratio between parts which didn't allow us to test the robot on paper. This ratio indifference caused several designs issues which was difficult to predict. After building the prototype, we realised that the main issue was stability and strength and decided to start designing a second prototype, which would have more stable legs. After brainstorming solutions to the stability and strength issue we concluded that the most reliable solution would be a greater surface area with standoffs between the two sides.

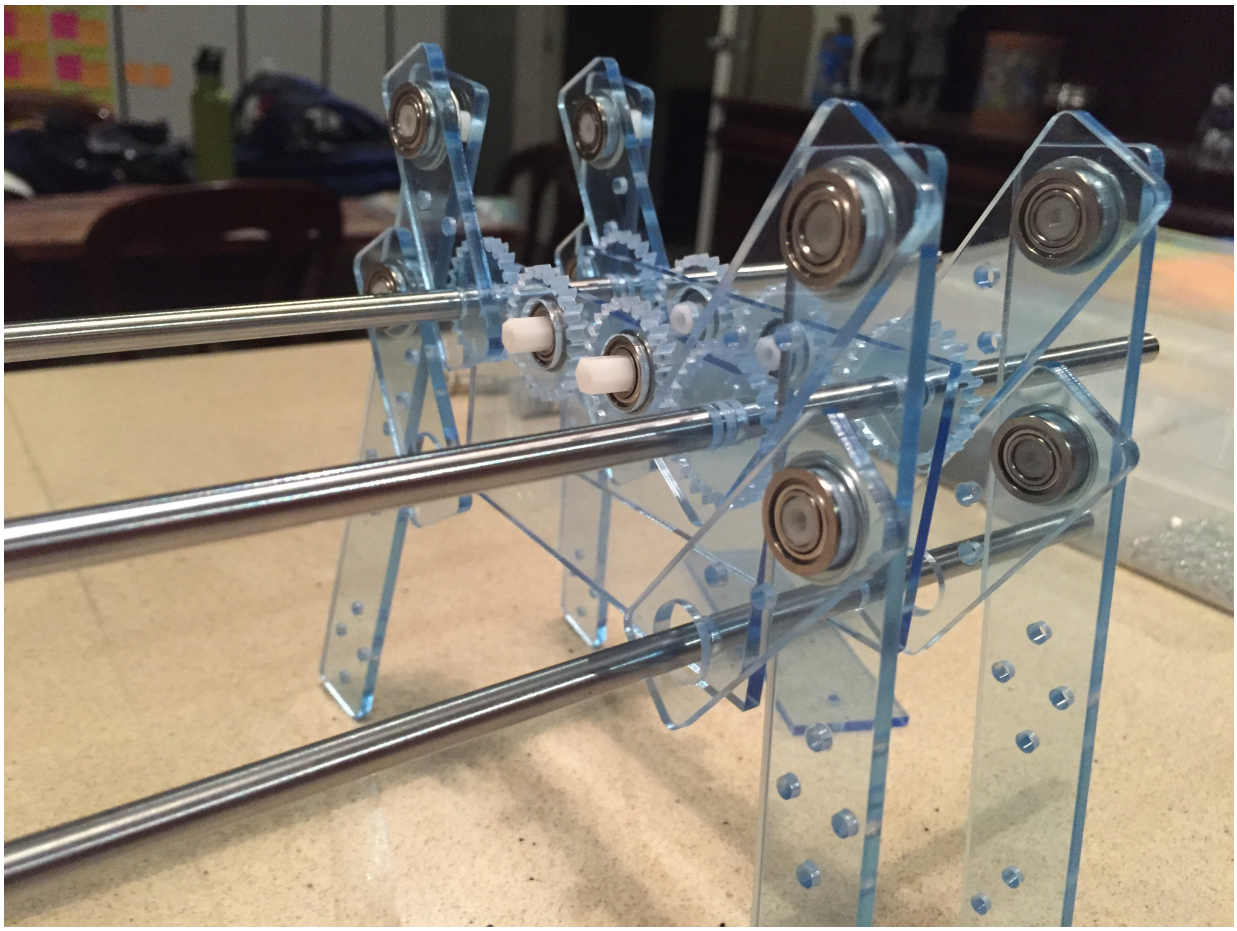
Issues/Solutions:

HARDWARE:

- stability / strength

SOLUTIONS:

- building new prototype with more stable using our brainstormed solution



Date: 15th March, 2016

Phase of Design Process: Prototyping

Agenda:

- Constructing and building prototype 2

Progress/Status:

1. DESIGNING AND BUILDING NEW PROTOTYPE (PROTOTYPE 2)

The second prototype was based greatly on the design of the first. The two side legs were added to the design completing the mechanism. The design was laser cut on yellow acrylic to differentiate it clearly from the first prototype. The main issue faced during the designing and building phase of this prototype was gearing the construction correctly to ensure the required movement. However, we realised that the design had some flaws with its design including ratios that were different, gears were slightly smaller than desired which would allow the legs to become out of sync and the body was missing holes. Realistically we concluded that the design was over engineered for the comp thus we will decrease its ability which

would hopefully reduce the height which is at a discomforting 18cm not including motors or wheels. However, the legs are longer then needed.

Issues/Solutions:

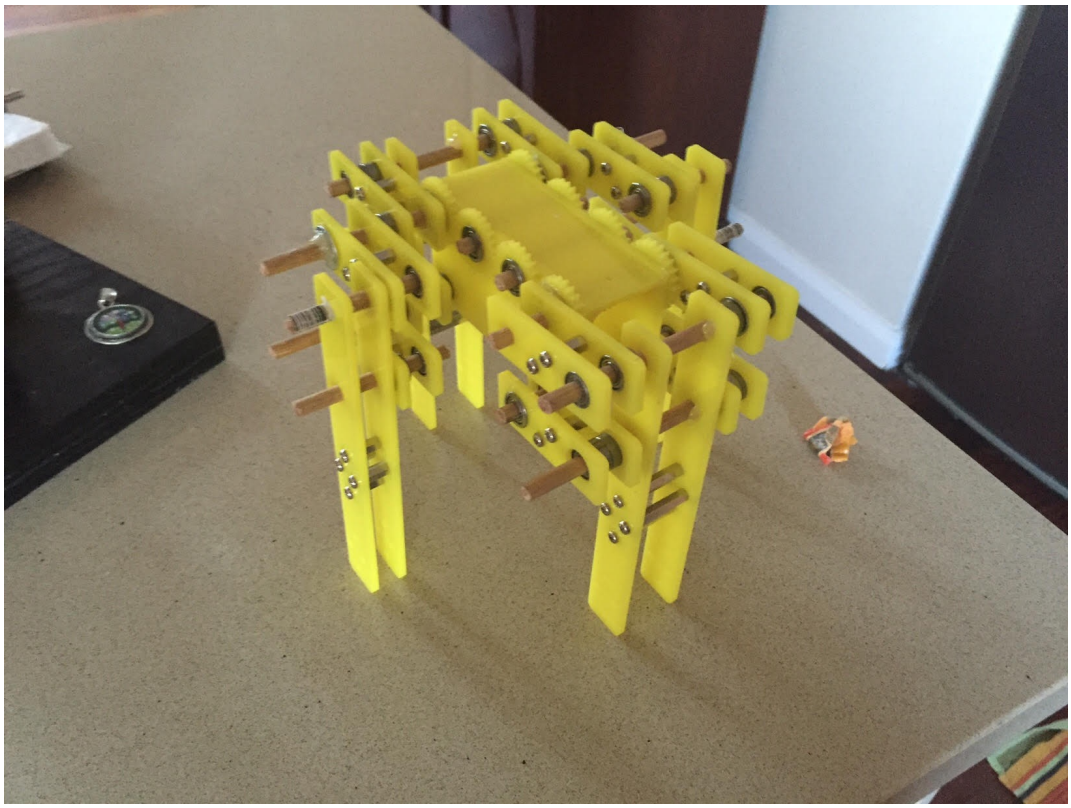
HARDWARE:

- The movement which was created by the mechanism was too great for the Rescue maze and would be over engineered for the challenge

-Missing elements to the design

Solutions:

- design and build new prototype that is more stable



Date: 15th March, 2016

Phase of Design Process: Prototyping

Agenda:

- Constructing the motor braces

Progress/Status:

1. DESIGNING AND BUILDING

At this point the robot had no means of rotating within the maze which would limit us to Mechanim wheels. We decided that these wheels although useful and proven to work in other situations would ultimately fail as our leg configuration wouldn't allow for efficient turning. We brainstormed ideas which would allow us to turn inside a 30cm x 30cm tile and decided upon a swivel wheel controlled by a Dynamixal Servo. Rory was quick to design it and 3D printed it by the end of the day. We did a quick and dirty job to allow us to test the initial idea and to approve on it (We call it a Dirty Prototype). This rapid prototyping is accentual to any idea and we believe that its crucial to innovative building of anything. This

process allowed us to prove that the idea wouldn't work which made us go back to the drawing board. We ended the day with the idea of using Omni Wheels.

Issues/Solutions:

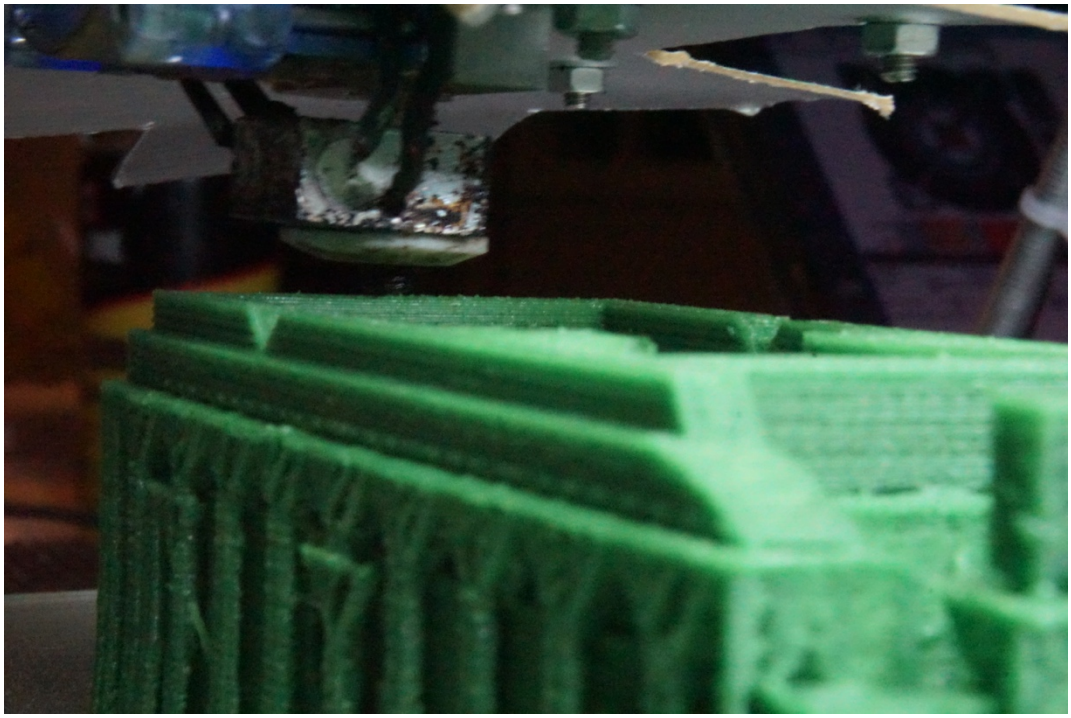
HARDWARE:

- The logistics of a swivel wheel

-Lack of ideas

Solutions:

- Design and rapid prototype ideas that we brainstormed



Date: 4th April, 2016

Phase of Design Process: Prototyping

Agenda:

- Designing and Building Fourth Prototype

Progress/Status:

1. THIRD PROTOTYPE

We built the next prototype of the robot and have realised that it is getting closer to what we would like the final prototype to look like and achieve. The gearing works much better and the ratio seems to be nearly correct. We have tested the mechanism several times and decided that it would be beneficial to create a final prototype, which would be more lasting. The last literation of the motor brace was designed, we all agreed that Omni wheels was the best solution for the issue. These parts were 3D printed days before and fitted perfectly with the Laser Cut parts. We are still waiting for the Dynamixal to arrive, though, but this has not However, we realised that we might have slight problems with the communication between boards (OpenCM, Teensy 3.2 and Intel Edison), but will start testing and try overcoming these possible issues.

Issues/Solutions:

HARDWARE:

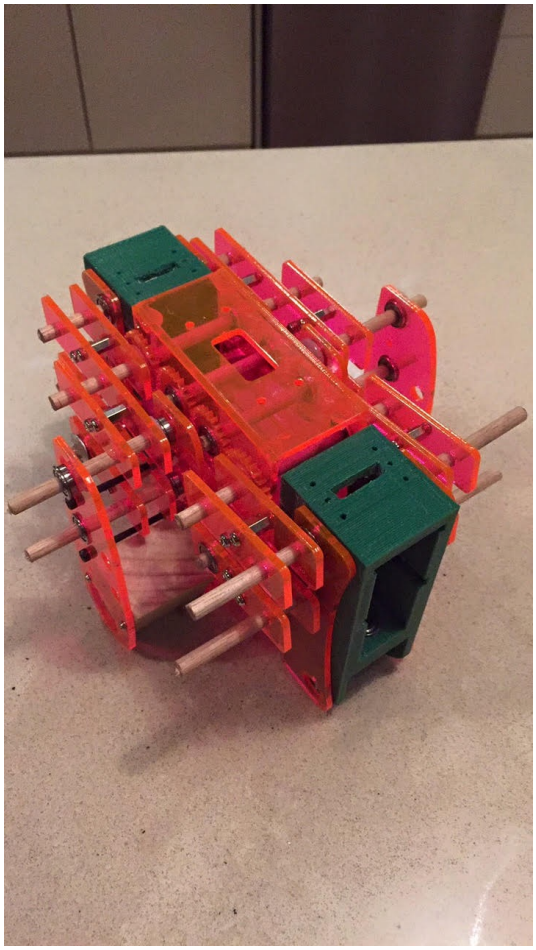
- The 3D printed and Laser cut parts have been designed to fit the Servo Motors specifications - they will most likely fit

SOFTWARE:

- possibility of an issue with the communication between the boards

Solutions:

- Wait for Australia post to pick their game up



Date: 20th April, 2016

Phase of Design Process: Prototyping

Agenda:

- Build and design fourth prototype

Progress/Status:

1. BUILDING FOURTH PROTOTYPE

To improve the design of the 4th prototype, we changed the ratio of the gearing and the legs, to increase stability. It was especially difficult to figure out the correct ratio for the legs, so we added some other drafts of components to advance the design towards the final product. This helped us visualise what it would look like,

mechanically, and we decided that there was still room for improvement through increasing stability and once again changing the ratio. Nevertheless, we realised that we were getting quite close to the finished design.

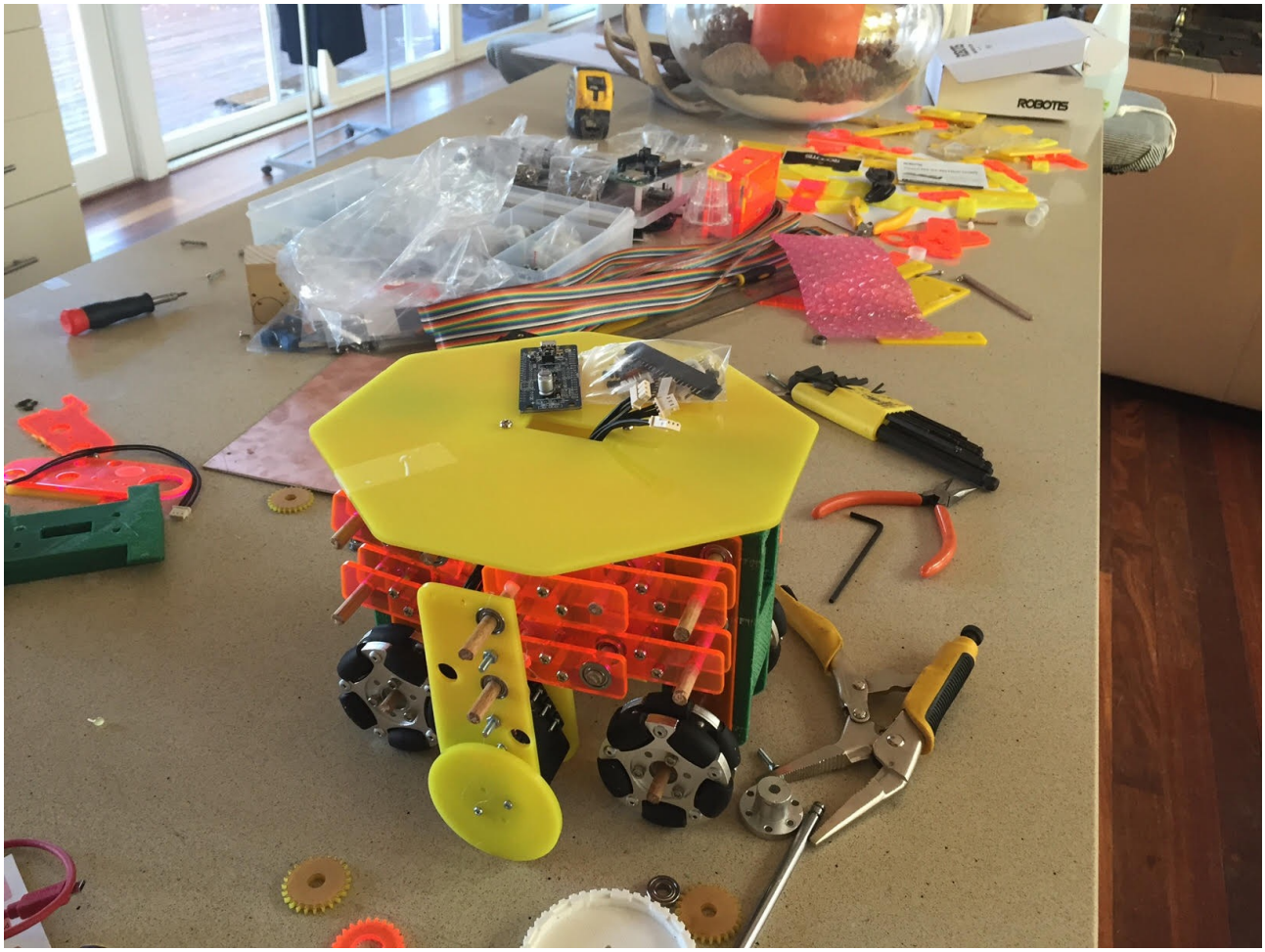
Issues/Solutions:

HARDWARE:

- stability

Solutions:

- Design new prototype with better stability - thicker legs and always having four contact points on the ground



Date: 11th May, 2016

Phase of Design Process: Prototyping

Agenda:

- Designing and Building Final Prototype

Progress/Status:

1. DESIGNING FINAL PROTOTYPE

We have finally built our final prototype. It has the ideal ratio and stability needed for the task and we have tested it vigorously. We have decided that this is the final prototype because it has been tested and works very well - a considerable improvement to all the previous designs. As can be seen in the picture below, we attached the LIDAR to the prototype and

ran a few sensor tests to check how it was working. We still need to start getting the robot to move around a maze, but the motors work very well and we think that it should be easier to program them than it was last year, especially since we now have far more experience. Its become extremely obvious that the robot has been over engineered for a 3cm obstacle so we decided to limit the movement of the robot in the final version with tabs.

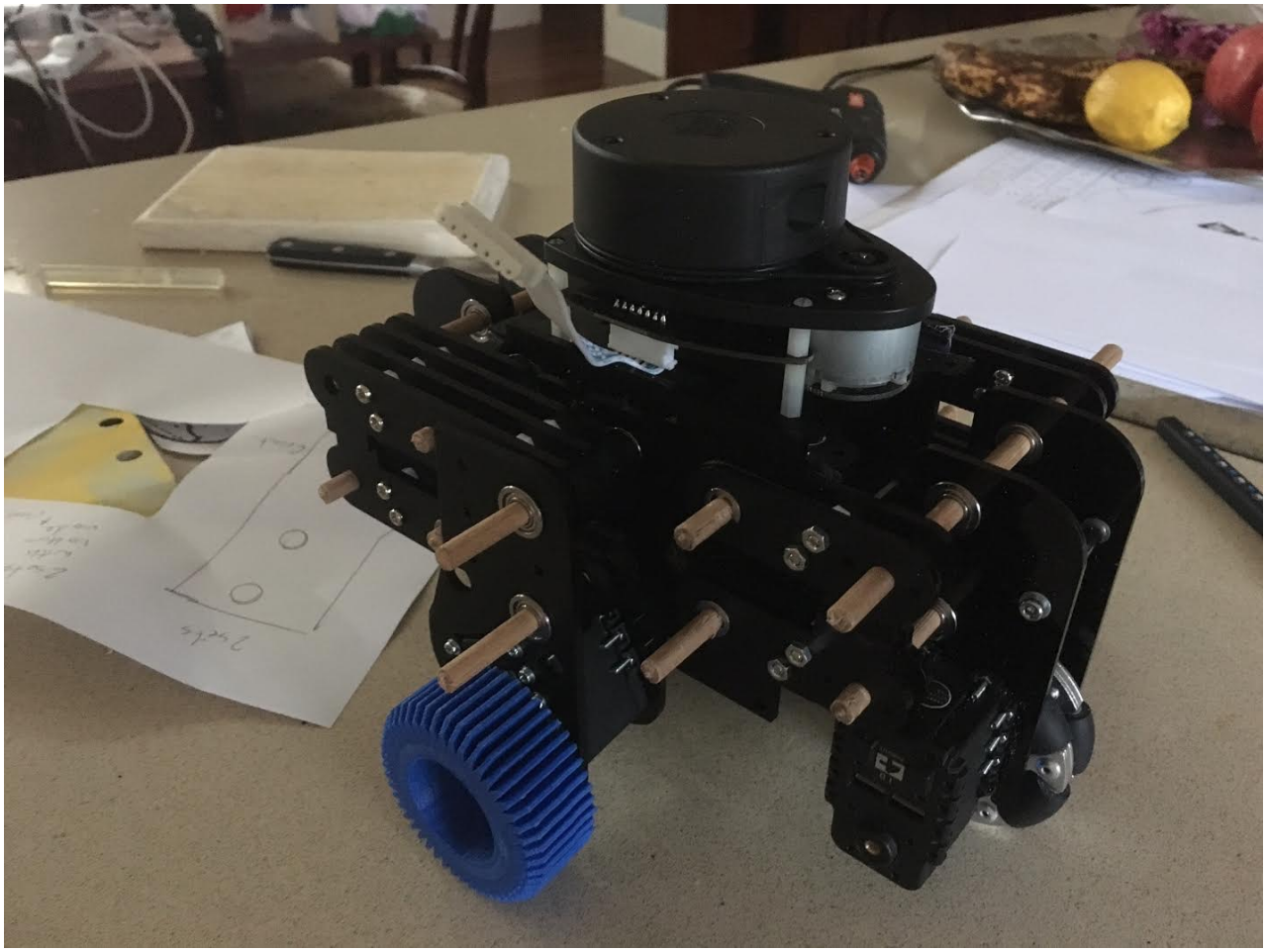
Issues/Solutions:

HARDWARE:

- Tabs are needed to limit the movement

Solutions:

- Change the Adobe Illustrator file



Date: 26th April, 2016

Phase of Design Process:

Agenda:

- Develop priority list for programming different components of the robot
- Make comprehensive list of communications needed between boards (Edison - Teensy - OpenCM)

Progress/Status:

1. PRIORITY LIST FOR PROGRAMMING

A mind map/priority list was developed for each of the different boards. Each board was ranked in importance to program and the programming components within each board were also prioritised. We determined that the most important board to focus on was the OpenCM, which will control the motors and therefore the robot's decisions. We decided that the next action would be to test once more the heat sensors and the drop mechanism, which worked well last year, and we will therefore be keeping the same ones. After that we deemed that it was important to program the odometry of the robot and finally, focus on SLAM and the rest of the sensors on the Teensy.

2. *LIST OF COMMUNICATIONS BETWEEN BOARDS*

We drafted a list of all the communications that will be needed between each of the boards. For instance, we will need to communicate between the OpenCm and the Edison for the odomtery.

3. *INCORPORATING SLAM - DISCUSSION*

We discussed the SLAM algorithm and how it would be possible to incorporate it into the program for the rescue robot. The SLAM algorithm uses landmark extraction for localisation, which could prove difficult for the robot inside the maze where there are not many landmarks to rely on. We brainstormed ways of using the SLAM algorithm for mapping without using landmark extraction.

Issues/Solutions:

SOFTWARE:

- may be hard to program SLAM without the landmark extractions

Solutions:

- research to figure out how to overcome SLAM's use of landmarks and see if the teensy / edison is even able to handle the stress of the SLAM algorithm

Date: 3rd May, 2016

Phase of Design Process:

Agenda:

- learning to program motors on OpenCM (Robotis)

- communication between boards

decided to change from using Teensy to just OpenCM and Edison

Progress/Status:

1. PROGRAMMING ON OPENCM (ROBOTIS)

We started programming on the OpenCM (Robotis) and found that it was almost identical to Arduino but has missing functions which would be nice to know about before we bought the board. The program itself was not hard to code, but it took some time to test all the Dynamixels and label them. We started writing the code for the Dynamixels and for the robot movements.

2. COMMUNICATION BETWEEN BOARDS

We found that it was increasingly difficult to gain communication between the OpenCM, Teensy and Edison, so we have decided to solely use the OpenCM and Edison for the robot. It was challenging to incorporate the Slave-Master algorithm for three boards, which would all have to 'talk' to each other. This means that both the mapping and the sensor codes will be on the Edison while the motors will be controlled by the OpenCM. We think that it will be easier to manage just these two boards, rather than including the Teensy as well.

Issues/Solutions:

HARDWARE:

- no issues

SOFTWARE:

- challenging to communicate using Master-Slave algorithm between three boards

Solutions:

- we will use only two boards: the OpenCM and the Intel Edison

Date: 10th May, 2016

Phase of Design Process:

Agenda:

- Design PCB Boards – Mini-Breakout Edsion / Open CM / I^2C Bus / Power Board / Power Level shifter

Progress/Status:

1. DESIGNING BOARDS

We spent the entirety of today designing every interaction between each board and how they will fit within the robot. One thing that we all decided was that the layout needed to be simple and modular. We decided that the smartest path for the robot is to be as simple as possible to allow as little issues as possible. One major issue which we faced today was a plan to communicate between an Edison and Open CM as they have different operation voltages. However, we found that the bigger Edison Arduino kit has an operation voltage of 5v or 3.3v perfect for the Open Cm but the Intel Edison Arduino Kit is huge.

Issues/Solutions:

HARDWARE:

- A lot to think about when designing these boards as they all rely on each other
- We predicted many things which my prove problematic later on.

Solutions:

- Measure twice cut once. We really need to spend more time designing the robots ecosystem and the interactions between each component.

Date: 10th May, 2016

Phase of Design Process:

Agenda:

- programmed heat to work
- OpenCM

Progress/Status:

1. PROGRAMMED HEAT SENSORS

We programmed the heat sensors to work for the Intel Edison, but were not immediately able to test them. The code is very similar to last year; as the sensors we are using are the same. We will be testing them shortly. The only problem we are expecting to face is that we have not properly labelled the circuits from last year and connecting them wrong could blow up the sensors, as has happened before.

1. PROGRAMMING DYNAMIXELS

We labelled the Dynamixals and started using the OpenCM to control the motor movements. We started by writing a simple pseudo code to plan out exactly what the motors had to do - each of the turns and turning around.

Issues/Solutions:

HARDWARE:

- no issues with the Dynamixels
- heat sensor could potentially blow up if plugged in wrong

Solutions:

- find circuit drawings for the heat sensors from last year, to make sure that we do not blow them up!

Date: 17th May, 2016

Phase of Design Process:

Agenda:

- programmed on OpenCM
- Slave - Master algorithm
- programmed the motors on OpenCM board

Progress/Status:

1. MOTOR CONTROL CODE

We continued programming the code for the Dynamixels. We tested them today to make sure that the motor movements were working. We have programmed the pseudo code we developed and have made sure that each of the turns works on just the motors.

2. SLAVE-MASTER ALGORITHM

We attempted to program the Slave-Master Algorithm using the OpenCM and an Arduino, and found that it was quite difficult as they used different libraries. We have continued trying, but have established that the algorithm does work between two Arduinos. It was decided that the problem was the OpenCM and we have started searching for ways of programming the Slave-Master Algorithm between the Arduino and the OpenCM to get past this barrier.

Issues/Solutions:

HARDWARE:

- none

SOFTWARE:

-communication between the OpenCM and the Arduino (not similar libraries)

Solutions:

- scour the Internet for ways of solving this problem.

Date: 21st May, 2016

Phase of Design Process:

Agenda:

- get robot moving
- start working on Presentation
- mapping

Progress/Status:

1. GETTING THE ROBOT MOVING

We started working on getting the robot moving using the Dynamixels. The robot moved efficiently, but it was difficult to get it to move through the maze. The LIDAR was not able to communicate with the Edison or the OpenCM as it talks in binary instead of bytes or the Edison is emulated incorrectly maybe with timing and we realised there were huge problems with communication. So, just to get the robot moving through the maze, we substituted the LIDAR for ultrasonic sensors as back-ups. Nevertheless, we had trouble with the ultrasonics as well and we assumed this was because they had blown up, but it ended up being a problem in the code with reading the ultrasonics. After strenuous trying to connect the Edison to the OpenCm and the LIDAR, we decided that the OpenCM was at fault. However, the issue is that we would have to use it to power the Dynamixels. So, our next task is to figure out how to get past the necessity of using the OpenCM.

2. START WORKING ON THE PRESENTATION

Meanwhile, we have started working on the Presentation for our robot, which would describe its components as well as our design process. We have used a similar template to the one from last year, but have obviously started doing research for the newer components.

3. MAPPING

We have started the mapping algorithm and have decided to use Depth-First Search. We decided to use it because it is the simplest and most efficient way of ensuring that the robot explores the entire maze. The mapping algorithm has already been written in Swift, but we will have to transfer it to JavaScript to make it work with the Edison.

Issues/Solutions:

HARDWARE:

- ultrasonic sensors weren't working

SOFTWARE:

- cannot communicate with OpenCM

Solutions:

- problem in code with reading ultrasonics
- researching possible solutions of getting past using the OpenCM to control the motors

Date: 22th May, 2016

Phase of Design Process:

Agenda:

- Testing new method of controlling the Dynamixels
- Continuing work on the Presentation
- Continuing work on Mapping
- Programming heat sensors on Teesny

Progress/Status:

1. NEW METHOD OF CONTROLLING THE DYNAMIXELS

We managed to find a circuit, using an Analog Tri-State Buffer, which will control the Dynamixels without using the OpenCM, so we have finally been able to scrap it! The circuit was extremely difficult to find as smart people know that mixing TX and RX on a single wire isn't smart due to the precision needed to communicate effectively. The circuit buffers data from the TX and RX respectively until a control pin is toggled which allows is to talk Half-Duplex to the Dynamixal. We now only have to get communication working between two boards: the Teensy and the Edison. We have gone back to the Teensy because we realised it works so much better and its teensy (tiny).

2. MAPPING

We continued working on the mapping algorithm and tried solving some bugs in it. It was also a challenge to connect from the Mapping algorithm on the Edison to the rest of the code on the Teensy. This will be our focus for the next couple workshops.

1. PROGRAMMING THE HEAT SENSOR

We tried programming the heat sensor on the Teensy, but read the circuits wrong, as they were covered by some hot glue. The heat sensor blew up, and we couldn't use it again. We managed to find another heat sensor to test the code on later.

Issues/Solutions:

HARDWARE:

- heat sensor broken

SOFTWARE:

- difficult to communicate between the Teensy and the Edison

Solutions:

- found a new heat sensor
- continue with trial and error and try to get communication going. Found a solution online which uses the emulated Arduino side of the Edison to save data to files which it receives from the TX-RX pin. \

Date: 24th May, 2016

Phase of Design Process: Production

Agenda:

- Finalising the PCB Boards
- Redesign Teensy Board with the new method of moving Dynamixal

Progress/Status:

1. FINALISING PCB BOARDS

The PCB boards were designed, etched and assembled.

1. PROGRAMMING THE LIGHT SENSOR

We also programmed the light sensor and have set the thresholds for all the different colours we will have to detect (white, silver and black).

1. MAPPING AND COMMUNICATION WITH TEENSY

We have continued fixing the mapping algorithm and have fixed some of the bugs. It has still provide difficult to communicate between the Edison and the Teensy, but we will continue trying.

Issues/Solutions:

HARDWARE:

- no issues

SOFTWARE:

- persistent issue of communication between the Edison and the Teensy

Solutions:

- continue with trial and error

Date: 25th May, 2016

Phase of Design Process: Testing and Implementation

Agenda:

- Heat sensor
- Mapping
- continuing documentation

Progress/Status:

1. PROGRAMMING THE HEAT SENSOR

We have managed to program the heat sensor to work and detect heat. Now, all there is left in regards to the heat sensor is to build it into the robot. We also have to incorporate the drop-mechanism code into the heat sensor and the mapping code and join them together.

1. PROGRAMMING THE LIGHT SENSOR

We also programmed the light sensor and have set the thresholds for all the different colours we will have to detect (white, silver and black).

1. MAPPING AND COMMUNICATION WITH TEENSY

We have continued fixing the mapping algorithm and have fixed some of the bugs. It has still provide difficult to communicate between the Edison and the Teensy, but we will continue trying.

Issues/Solutions:

HARDWARE:

- no issues

SOFTWARE:

- persistent issue of communication between the Edison and the Teensy

Solutions:

- continue with trial and error

Date: 28th May, 2016

Phase of Design Process: Testing and Implementation

Agenda:

- getting the robot moving through the maze
- mapping

Progress/Status:

1. GETTING THE ROBOT TO MOVE THROUGH THE MAZE

We have managed to get the robot moving slightly through the maze, following a very basic PID algorithm, but as soon as we connected it to the Mapping, the PID was confused. So, we have been focusing on getting the robot to move properly through the mapping algorithm.

2. MAPPING

We have finally achieved communication between the Edison and the Teensy, but it is still proving difficult to communicate fully between the mapping algorithm and the rest of the code. We will continue working on this and have placed it as the main priority to fix it.

Issues/Solutions:

HARDWARE:

- no issues

SOFTWARE:

- issues with mapping and PID not being able to work simultaneously - inefficient communication

Solutions:

- we have devised a detailed protocol to help with standardising the communication between the Edison and the Teensy, which will hopefully help with getting communication working.

Date: 4th June, 2016

Phase of Design Process:

Agenda:

- get communication between devices

Progress/Status:

1. COMMUNICATION BETWEEN BOARDS

We have managed to use the protocol to get some communication between the boards but the robot is still getting confused while travelling through the maze. We need to work on giving the robot better instructions from each of the boards.

Issues/Solutions:

HARDWARE:

SOFTWARE:

- communication between the boards is confusing the robot

Solutions:

- keep trying to perfect the communication. meanwhile - continue working on programming the other sensors such as the accelerometer.

Date: 11th June, 2016

Phase of Design Process: Testing and Implementation

Agenda:

- program accelerometer
- communication between boards

Progress/Status:

1. PROGRAM ACCELEROMETER

We managed to program the accelerometer to work and will soon incorporate it into the code. It will allow the robot to know how much it has turned and if it has truly turned 90º. We think that this will be greatly beneficial in the maze navigation and especially with the mapping algorithm and PID.

2. COMMUNICATION BETWEEN BOARDS

We have started trying to give the robot instructions while moving around the maze, and were having trouble determining which parts of the code the robot was messing up on. So, we started debugging the robot to the screens and making sure the protocol was being followed. We found the errors that were occurring and then decided to change small sections of the code. We realised that the mapping had a bug and had to be fixed, and the PID algorithm was not correctly tuned. This is a task to fix for next/our last workshop.

Issues/Solutions:

SOFTWARE:

- communication between boards is confusing robot at times

Solutions:

- we will try fixing the mapping algorithm and the PID algorithm to help the robot move around the maze.

Date: 18th June, 2016

Phase of Design Process: Testing and Implementation

Agenda:

- final workshop

- testing the robot and getting the mapping working

Progress/Status:

1. GETTING THE COMMUNICATION AND MAPPING WORKING

We have FINALLY managed to get the mapping working with the Teensy code! The robot was able to move around a homemade maze and search ever section of it. However, it did not do it in the most efficient way possible, so our challenge (if there is time) is to get a more efficient mapping algorithm. This might be something to fix for next year's robot, though as one of our team members is leaving to Germany very soon and the competition is only a few days away. We still have to incorporate all the sensors into the robot and include the sensor code into the main code. Finally, we have to tune PID, which hasn't been calibrated in the best possible way, which we will be doing as soon as possible.

Issues/Solutions:

HARDWARE:

- none

SOFTWARE:

- mapping algorithm is not as efficient as it should be
- PID is not calibrated

Solutions:

- we will fix the mapping algorithm if there is time

- we will tune PID to make it more effective

Date: 20th June, 2016

Phase of Design Process: Final changes

Agenda:

- Locating out any bugs that might be within the code

Progress/Status:

1. GETTING THE COMMUNICATION AND MAPPING WORKING

The code on the teensy although it worked had many bugs which would reduce the accuracy of the mapping thus needed fixing. The first issue was hardware with the I^2C bus giving the Teensy invalid readings making the robot spin indefinably. PID also had slight issues with its settings, the robot was set to look at 22.5 Degrees (from 90 & 270) which would cause the robot to be too predictive of its future and ultimately crash into walls but with a simple fix PID was tuned much better using Ziegler-Nichols' method. The method uses variables like Oscillation time and loop speed to calculate KP correctly which then allows the KI and KD to be tuned. As the IMU uses its own microprocessor to gather data we learned that we needed to wait a set amount of time to allow the sensor to reboot and reset.

Issues/Solutions:

HARDWARE:

- small issue with I2C board (feedback noise causing IMU to give invalid readings)

SOFTWARE:

- mapping algorithm is not as efficient as it should be
- PID is not calibrated

Solutions:

- we will fix the mapping algorithm if there is time
- we will tune PID to make it more effective