THESEUS

The Greek hero that chose to enter the labyrinth.

A cost effective, tele-operated observation search and rescue robot.

Presented by:
Jordan Alan Haskel

Prepared for:
Justin Pead
Dept. of Electrical and Electronics Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Mechatronics

THESEUS: A COST EFFECTIVE WHEELED RESCUE ROBOT SOLUTION

Project in partial fulfillment of the requirements for the degree of

Bsc(Eng)
Mechatronics

Univeristy Of Cape Town
Engineering and the Built Environment

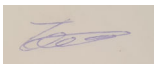
Approved by

Supervisor: Justin Pead

[November 10, 2017]

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature: 
J. A. Haskel

Date: November 10, 2017

Abstract

Urban search and rescue workers are often put in a position to enter dangerous situations to find survivors. In many cases the lives of search and rescue workers are lost in order to investigate environments where there are no victims to be found. This project aims to reduce the need to risk peoples lives in order to investigate urban search and rescue situations through the use of robotics. Rescue robots already exist and previous designs have been built and tested. However, rescue robotics is a relatively new field in the world of technology and has room for many improvements. Rescue robots aim to provide the operators with information about the terrain without concerns about the difficulty of overcoming obstacles or damaging expensive equipment. Current robot designs are often expensive and either large in order to overcome obstacles (reducing ability to fit in small spaces) or small in order to fit in small spaces (reducing climbing ability). Theseus is aimed at proving that it is possible to design and manufacture a cost effective, tele-operated search and rescue robot that is capable of fitting into small spaces and overcoming obstacles.

The project makes use of various engineering methods (computer aided drawing, simulation software and laser cutting) and design approaches (iterative refinement, calculation, debugging and simulation) which follow pre-structured design plans in order to achieve the goal at hand. The design also makes use of previous attempts at cost effective rescue robot designs [T. J. Mathew, "Scarab: Development of a rugged, low cost, inspection-class robotic platform", Master's thesis, 2015.][M. Wilson, Development of a low-cost, mid-sized, tele-operated, wheeled robot for rescue reconnaissance, 2013.].
Theseus is designed to be modular and expandable in order to investigate various solutions to specific issues and determine which solutions work best and why. The design makes use of mechanical design solutions such as Load Intuitive Modules (LIMs) for driving and climbing, as well as electronic solutions such a RC communication, accelerometers and video transmission.

The study includes testing of the system in a reliable and repeatable manner. The tests carried out are to be used to determine the effectiveness of the various designs used, as well as where future iterations could improve. The system proved effective in it's aims of being low cost and tele-operated. The design showed promising proof that the LIMs were capable of achieving the desired outcomes however the system had issues relating to robustness which would need to be solved before Theseus could be used as a rescue robot.

Acknowledgements

This project was given with the requirement of being completed by a single student. However I can say with absolute certainty that Without the help and support of certain people completion of this project would never have been possible. I am deeply grateful for all the help, support and motivation that I received from those around me.
In particular I would like to thank:

Justin Pead: For the endless help and guidance and encouragement to keep trying when things did not go as planned. I will never be able to express my gratitude for the time that you gave that allowed me to work on a project that sparked my interest.

My Family: For the endless love and support in every sense of the word. Thank you for helping get to this point in my life.
Thank you mom for always encouraging me to do my best and reminding me that my best is all I can do.
Thank you dad for keeping me going and reminding me to stop and enjoy the sunsets along the way.

Levi: Destined to struggle and succeed side by side since day one.

Gina: For the endless love, support and comforting me when I needed it most. Thank you for reminding that everything will be ok (and that I need to eat and sleep.)

Brendan: For the countless early mornings and the valuable advice.

Ross, Chris and Kev: For the endless mechanical advice and help.

Jason: For the assistance in all of my bizarre requests.

Jose C: For the much needed pick me ups.

Table of Contents

List of Tables

List of Figures

1 Introduction

This report will go through the process of designing and building a cost effective robot for aiding those that work in the field of USAR (Urban Search And Rescue). The robot will have the express purpose of helping to keep search and rescue workers out of harms way while providing the desired reconnaissance.

1.1 Background of the Project

Tragic events occur all too often in urban environments these could range anywhere from house fires to collapsing high rise buildings or city wide floods. Owing to the dense populations in urban areas these events put the lives of many people at risk. In order to curb the destruction and loss of life brave search and rescue workers put their lives on the line in the attempt to help others. Finding survivors in such environments puts search and rescue workers lives at risk. In order to reduce the risk to search and rescue workers the use of observation robots could prove highly effective. Observation robots can enter perilous environments, search for survivors and assess the safety of the situation without and further risk to human life. Once these robots have found any survivors and investigated the safety of the situation, search and rescue workers can strategically enter the environment in a way that is least risky to themselves and the survivors. This allows search and rescue workers to extract survivors and exit the area in a shorter time.

1.2 Project Objectives

The main objectives of this project are:

1. Understand the requirements of the project.
   This required an understanding of what a search and rescue robot would encounter and the means to construct a system that can handle the tasks to be encountered.
2. Conduct a literature review of previous work in this field and critically evaluate current technology/research
3. Design a "small cost effective remote observation vehicle" capable of maneuvering though an urban building environment the scope was thus expanded during the course of the project and the final requirements included:
   - Climbing obstacles.
   - A suitable battery life.
   - A simple to use tele-operated remote control system.
   - A video feed system.
   - Simple orientation sensing.
4. Test and compare the vehicle capabilities.

These objectives have been adapted from those given in the formal project description as shown in appendix A.

1.3 Scope and Limitations

The formal scope of the project can be seen in appendix A. The description of the project is stated as:
" Disaster environments are dirty and dangerous, which makes the role of rescue aid ideal for a robot. The main role of a rescue robot is to provide beneficial information to the rescuers without burdening the crew with regard to environment difficulties or concern of the wellbeing of their device. Larger robots could struggle to navigate the terrain or can be costly to purchase or maintain. Small low-cost platforms often do not have these capabilities to survive the environment and provide no value to a rescue team. Alternative approaches were tested (T. J. Mathew, 2015; M.Wilson, 2013) and provided plausible systems to the costly larger robots.
This project will investigate a hybrid design between the earlier projects conducted at UCT. The focus of this project will be on overcoming obstacles that may be found in a USAR environment with a strong focus on off the shelf and cost effective solutions to reduce concerns over vehicles in disaster zones."
The scope of the project is limited to a single system that is concerned with overcoming common obstacles. The system is to be a hybrid adaptation of those tested before.

External constraints on the project:

- 10 weeks were given for the research, design, implementation, testing and documentation of the project.
- The project is to be carried out by a single student.
- A budget of R1000 was allocated for the purchase of necessary equipment and materials as approved by the students supervisor.

1.4 Plan of Development

The project is split into 6 distinct phases namely:

1. Research and literature review
2. Design planning
3. Design, calculation and simulation.
4. Fabrication, manufacture and implementation
5. Testing
6. Analysis of results and conclusion

A rough idea of the time that would be spent for the activities that would satisfy the phases can be seen in the Gantt chart in figure 1.1. This schedule realistically provided extra time for any problems or corrections that would need to be made.

projected gantt chart			DURATION (days)
START DATE	END DATE	DESCRIPTION	
30-Aug	8-Nov	Whole Project	68
30-Aug	14-Sep	Literature review	14
14-Sep	15-Sep	Specification decisions	1
15-Sep	10-Oct	Mechanical design and implementation	25
25-Sep	15-Oct	Electronic design and implementation	20
15-Oct	24-Oct	Troubleshooting	9
22-Oct	31-Oct	Testing	9
31-Oct	8-Nov	Final documentation	8
30-Aug	31-Oct	Documentation	60

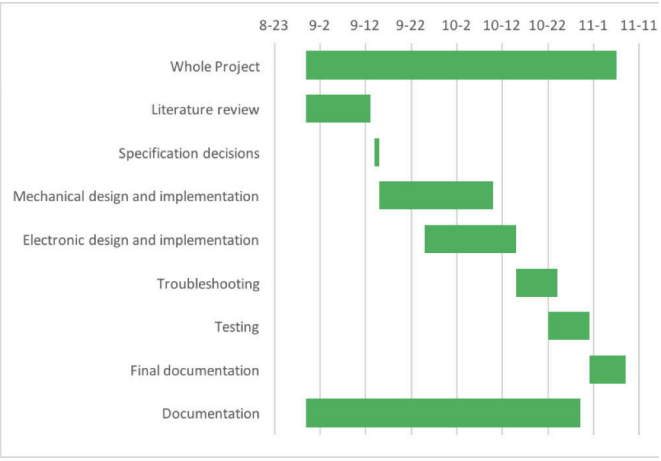


Figure 1.1: Projected Gantt chart for duration of project

1.5 Layout of the Report

The report consists of 9 chapters. Each chapter deals with a specific section of the project.

Chapter 2: Literature Review
This chapter provides background information of the subject of the report as well as the basics of some of the technical engineering practices or methods to be used.

Chapter 3: Specifications
The specifications are presented. These specifications are decided upon based on the expected requirements of rescue robotics as interpreted from the information in the literature review.

Chapter 4: Methodology
Steps, procedures and methods used in the design and implementation of the project are stated and explained in this chapter. This highlights the specific procedures that were chosen to be appropriate for various phases and tasks.

Chapter 5: Design Planning .
This chapter sets out the planning of the design phase. The practices, procedures and philosophies that will be used are explained.

Chapter 6: Design
This chapter in concerned with the calculation, design and simulation (confirmation) of the various sub systems to be put in place. This section also deals with the fabrication, manufacture and implementation of each sub system as well as the implementation into the final larger system.

Chapter 7: Test Procedure and Results
The procedure used to conduct tests is explained and the results of the tests conducted are presented.

Chapter 8: Discussion, Analysis and Test Conclusion
Deals with the analysis and discussion of the test results in order to find meaningful information from the results. The discussion and analysis of each test if followed by the conclusion based on the information provided by the test.

Chapter 9: Project Conclusions
The conclusions drawn from the project are given. A conclusion as to whether the specifications were met is presented, followed by a discussion of the notable limitations of the project. This section then concludes with recommendations for further work.

2 Literature Review

2.1 Background of Rescue Robotics

Research into the field of rescue robotics was necessary in order to ensure that the design process in this project created a useful and required solution to a problem. It would be redundant to recreate old solutions or solve problems that do not exist. Research into the topic helps gain ideas from the successful previous designs and allows for improvement on failed designs. research into the conditions of disaster sites is necessary in order to make informed design choices. This is to provide insight into the resources that would prove to be a useful tool in aiding the finding and rescuing of people in the case of a life threatening event.

2.1.1 History of Rescue Robotics

The Beginnings
In the beginning of the 20th century robots were not a popular part of science fiction yet, as explained by a paper about the history of robots [1]. The idea of robots only came around when the Capek brothers, Joseph and Karel started to write about them around 1917 [2], [3]. The Karel brothers are credited with the start of the word robot. The word originating from the Czech word *robota* meaning serf. It was only later that the concept of robots was further popularized by Isaac Asimov. Asimov introduced the term robotics after which his three famous laws to guide his fictitious robots' behaviors became well known. The laws being as follows[4]:

- "A robot may not injure a human being or, through inaction, allow a human being to come to harm."
- "A robot must obey the orders given it by human beings except where such orders would conflict with the First Law."
- "A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws."

Despite the science fiction setting and the application to fully autonomous robots. These laws represent a good fundamental guideline of what a rescue robot should achieve.

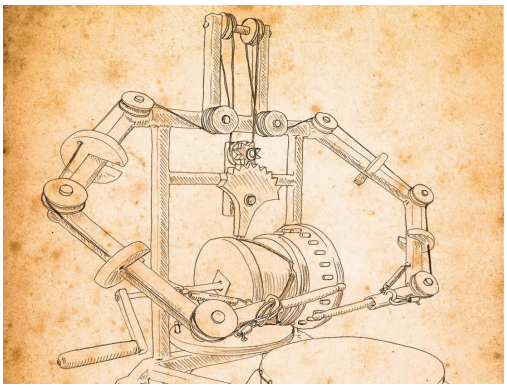
Imagining Robots

Even though the term robot only appeared as recently as 100 years ago people have been attempting to design and build machines to do their bidding for thousands of years. Rocco, Albo and Coelho explain that these include [5]:

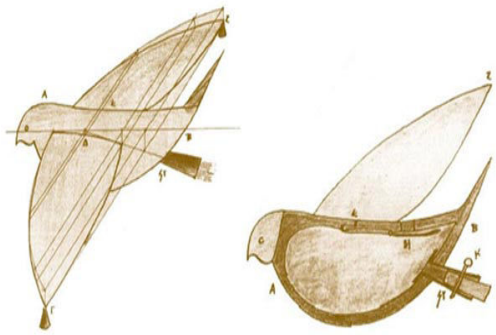
An artificer by the name of Yan Shi that lived 1023–957 BC in China. Yan Shi is said to have created a life sized "automaton" that moved in a "like for like manner" and could sing.

Archytas of Tarentum lived in the fourth century and is considered the father of mechanical engineering. He designed a mechanical steam operated pigeon.

Leonardo Da Vinci then created a robotic knight around 1495. Humans have been designing extraordinary robots throughout history and in the past century more has been accomplished than ever thought possible.



(a) Da Vinci Knight Robot



(b) Pigeon by Archytas

Figure 2.1: Inventions Leading to Robotics

Rescue Robots

Unfortunately tragedy and disaster have been the catalyst that has sped up the progress in the field of rescue robotics. There are many events that have strongly impacted the field of rescue robotics. Some of these tragic events are detailed below [6].

Kobe and Osaka Earthquake

At 5:46am on the 17th of January 1995, a 7.2 earthquake hit south central Japan killing 5 100 people and injuring 26 800 others, this earthquake also caused about 100 billion USD worth of damage. This tragic event drove Japanese interest in rescue robotics, including robots developed by the Tokyo Fire Department's Fire Science Laboratories.

Oklahoma City Bombing

At 9:03 am on the 19th of April 1995 Timothy McVeigh bombed the Alfred P. Murrah Federal building in Oklahoma City. He killed 168 people and injured over 500. John Blitch was one of the rescue workers that day, John was a Masters student in mathematical and computer sciences under Robin Murphy at the Colorado School of

Mines. Upon telling Robin of the ordeal, John and Robin committed to a new field of research and both of them were involved with the rescue efforts on 9/11. Blitch went on to become the program manager at DARPA (Defense Advanced Research Projects Agency) for the Tactical mobile robots program.

9/11 Manhattan Terror Attacks At 8:46 am on the 11^{th} of September 2001 a plane hit the north twin tower, 17 minutes later another plane hit the south twin tower. John Blitch and Robin Murphy the same colleagues mentioned above began organizing a rescue robot team at 9:15 that morning. Groups such as Foster-Miller, iRobot and the Navy robot lab: Space and Naval Warfare Systems Command (SPAWAR) aided in this effort to help. These groups assisted in finding 10 victims (more than 2% of victims found). The robots helped by providing:

- Light
- Sound
- Communication potential
- Color and infrared video
- Global positioning
- Mapping
- Sonar detection
- Biological detection
- Chemical detection

Many of the functions that rescue robots can provide relate to searching for signs of life. These signs of life are not a strict combination of things that must be found but any combinations of readings that a robotic system can take in order to ascertain whether there is a victim that is still alive. Some of these signs of life are requirements for testing courses for rescue robots and can include:

- Form (simulated with mannequins dressed as civilians and rescuers)
- Motion (simulated with waving arms and moving fingers)
- Body heat (simulated with heating pads and blankets)
- Sound (simulated with audio tape recorded shouting, moaning, tapping, and locator beacons)
- CO2 emission (simulated by being concentrated in voids supplied by nearby tanks)

These signs of life can be used together to determine a victims condition and conclude whether the victim is aware, semi conscious or unconscious [7]. The robots can also map these victims to an occupancy grid which allows all the victims to be seen on a map with reference to the environment [8].

2.1.2 The Need for Rescue Robotics

From the events described above we can see that the field of rescue robotics could play an integral role in saving the lives of disaster victims as well as reducing the risk to the lives of rescue workers. Robots have taken a large swing towards replacing people that work in fields that are described by the three D's:

- Dangerous
- Dirty
- Dull

Rescue attempts can be seen as dirty and definitely as dangerous. The number of sources and articles delving into the topic of rescue workers that have lost their lives is horrifying. In an article by Bryce Hall from 2013 [9], he states that more than 400 first responders lost their lives in the aftermath of the 9/11 attacks. The US Department of Labor's, Occupational Health and Safety Administration show that rescuer fatalities account for more than 60% of confined space fatalities [10]. There is a dangerous long term effect of chemicals and particulate matter inhaled by responders to the 9/11 attacks. This has caused suffering in the years following the rescue efforts[11].

2.2 Operating Conditions

In order to ensure that a robot will be able to deal with the environment it is placed in a short analysis of various disaster conditions needs to be done. In the following sections Urban search and rescue is explored as well as land, water and air robotic applications. From this analysis certain specifications and testing goals can be set, this will allow for a helpful and meaningful design process that should yield a useful product.

2.2.1 Urban Search and Rescue

Urban Search and Rescue (USAR) " involves the location, rescue (extrication), and initial medical stabilization of individuals trapped in confined spaces. Structural collapse is most often the cause for people being trapped, but individuals may also be trapped in transportation accidents, mines, and collapsed trenches." [12]. This is the definition given to urban search and rescue by The Federal Emergency Management Agency (FEMA).

Standard Search and Rescue Conditions

Unfortunately to state standard urban search and rescue conditions would be counter productive. The nature of such situations is inherently unpredictable. Certain disaster zones may pose widely different challenges that a robot would need to overcome. Nevertheless a specific metric is necessary in order to determine if a rescue robot is able to successfully overcome an object. This metric helps define what constitutes a success. A standardized test course for a rescue robots and the various ways in which they set metrics to determine success is available. However, this does not define the general size

of obstacles that will be encountered [13]. This is for good reason, each scenario will be different as different disaster sites will have greatly varying obstacle sizes depending on a plethora of variables. In this project it is decided that overcoming a standard step sized object would be a good metric for success. According to International building code for stair treads and risers the size of a standard step should be between 102mm and 178mm [14].

USAR Needs

FEMA and the National Institute of Justice (NIJ) co-sponsored an effort to identify and define certain functional requirements for desired technologies. These requirements aimed to meet the needs of USAR and law enforcement agencies. This effort involved many other organizations as well. In the report that resulted from this effort, high priority needs are listed. Some of the needs are given below [15]:

- "Improved real-time data access (data pertaining to site conditions, personnel accountability, medical information, etc.)"
- "The ability to accurately and non-invasively locate survivors following structural collapse – the ability to "see" through walls, smoke, debris, and obstacles"
- "The ability to communicate (transmit signals) through/around obstacles"
- "Lighter, more efficient power sources (batteries, fuel cells, or other technologies able to power multiple systems for longer periods of time)"
- "Improved monitoring systems (i.e., atmospheric, biomedical, personnel accountability, etc.) - real-time, portable, multi-function devices that expand on existing detection capabilities"
- "Reliable non-human, non-canine search and rescue systems - robust systems that combine enhanced canine/human search and rescue capabilities without existing weaknesses (i.e., robots)"

These needs can be used to help define the specifications for the project at hand.

2.2.2 Robots In Environments

Different environments have different demands on the designs of robots that must operate in them. These demands and how the robot handles them will determine how successful the robot is in achieving its primary objectives.

2.2.2.1 Water

Water disasters occur in many contexts, they can include anything from deep sea rescue to local flooding of towns or bridges. A robot suited to help with rescue in such situations would need to be highly adaptable to deal with variations from strong to no currents

in an energy efficient way. The risk of water entering the body of such a robot is also of great concern and would require a robust water proof design. Owing to the nature of water rescue, time is of essence as humans face the threat of drowning. For this reason the use of an observation robot would be less useful than it would be for other rescue applications. A small search robot will not be able to save the person itself and a rescue team would then need to go into the water once the robot has found a target. Thus the human rescuer is still in danger owing to the time sensitivity. Consideration of the time concern and the minimal reduction in risk to human rescuers, shows that water application may not be an effective field to explore for the use of a search and observation robot.

2.2.2.2 Air

Air rescue provides a relatively obstacle free environment for a robot. Airborne robots can survey terrains and environments without having to overcome many obstacles. It also allows for fast movement and broad views of large terrains or even small spaces, making it ideal for search and rescue. However, airborne robots face the problem of not being able to conserve battery while the operator makes decision or has to tend to other problems. This results in short battery life and highly time pressured situations.

2.2.2.3 Land

Land based robots can stay still while the operator makes decisions which helps preserve battery life. However, land rescue poses problems in the form of rubble and other obstructions which hinder smaller robots. Larger robots may have less problem with rubble and ground obstructions but they have a problem fitting into small gaps and low spaces. Thus a trade off between robustness and size occurs. Many such trade offs are apparent but one of the main trade offs for land based rescue robots is between legged robots and wheeled robots. Wheels are the preferable choice of locomotion for efficiency and simplicity, while legs are the preferable choice for the ability to traverse rough terrain [16]. Many new projects have begun to explore the possibility of merging the positive aspects of each of these classes. Most of these attempts make use of wheels attached to the "feet" of the robots legs. This is innovative however it often still involved the movement of the legs or a complex leg design [16], [17], [18], [19],.

2.3 Past and Current Designs for Rescue Robotics

Looking at past designs has been done to provide some insight into concepts that have been previously used. Those that work have been drawn upon for inspiration and those that have failed can be drawn upon for lessons or possible improvements. Looking at past and current projects can also be used to determine the cost range of current robots that are being utilized.

2.3.1 Current Cost of Robots

The current price of robots is an important factor to consider when deciding to design a "cost-effective" robot. The prices of current available robots allows one to determine a metric for what would be considered low cost. At first attempt pricing for rescue robotics is not readily available, this could be owing to the difficulty of actually procuring rescue robots that is attached to their rarity and price. Therefore the prices of simple non-rescue robots was investigated first. A robot called the TurtleBot which can be used as a robotic education tool and as a research and development robot was investigated. The TurtleBot3 has a cost ranging from 549 to 1,799 USD (7,086 to about 23,221 ZAR) depending on the model. These figures were found at the time of writing on sales sites that are linked to the the TurtleBot website [20] on which it is also stated that the “TurtleBot is the most affordable platform for educations and prototype research & developments.” In order to get more price estimates quotes were acquired for some rescue robots or robots that could do a similar job. A quote from Inuktun estimated that an observation robot, not intended for rescue operations, would cost around 35,000 USD (453,062 ZAR) and for them to build a custom robot would cost over 100,000 USD (1,294,465 ZAR). This email can be seen in appendix B. Upon further investigation into some academic papers and journal articles some Rescue Robot prices were obtained, These prices told a similar story. The Scarab report states that the cost for throwable robots is around 13,000 USD (168,280 ZAR). Tethered rescue robots made by USF Perceptual Robotics Laboratory in partnership with Inuktun cost 8,000 to 13,000 USD (103,557 to 168,280 ZAR)[6]. The same source states their untethered robots range from 33,000 to 40,000 USD (427,173 to 517,786 ZAR) while also stating that that fire rescue departments will only be able to consider a robot for less than 10,000USD (129 446 ZAR) and that 3,000USD (38,833 ZAR) is the most realistic budget.

2.3.2 The Scarab

The Scarab [21] is a project out of the University of Cape Town’s Robotics and Agents Research Lab (RARL). The scarab is a small sized robot, that has been designed to be throwable. The scarab can be seen in figure 2.2. This Robot was designed to be rugged and aimed to try deal with as many of the infinitely wide number of operating environments in USAR as possible. The robot was also an attempt to fit into small areas that had been previously unaccessible to traditionally large rescue robots. The design of the wheels was a focal component of the robot. The wheels as seen in figure 2.2 are large and impact absorbing this allows the robot to be thrown as the wheels being larger than the body will most likely absorb the

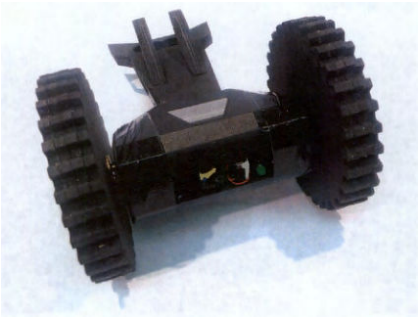


Figure 2.2: The Scarab by RARL.

impact. The large wheels also allow for climbing larger objects than small wheels. The project aimed to be low cost at less than 500 USD. 500USD was considered cheap enough to be expendable by the report. By comparing this figure with the realistic budget of the fire rescue departments discussed in the previous section the robot definitely meets a cost effective criteria however it is not quite at the stage of being deemed disposable if they can only purchase 6. A loss of one of six robots could possibly be cause for concern.

2.3.3 UCT Mechanical Project

Matthew Wilson a past mechanical engineer at the University of Cape Town, undertook a final year project in the field of rescue robotics [22]. The project was titled "Development of a Low-Cost, Mid-Sized, tele-Operated, Wheeled Robot for Rescue Reconnaissance." The design created in this thesis made use of Load Intuitive Modules (LIMs) shown in figure 2.3 which rotate the linkage holding the wheels so that the rear wheel can climb over the front wheel if it gets stuck. A sketch of the design robot design can be seen in figure 2.3 It also reduces the motor cost as one motor can drive two wheels as well as make them automatically flip when needed. This design would be better in the use of small robots than traditional wheel assemblies would be, as it would allow the robot to cope with much larger obstacles. This project aimed to cost R4000 or less and ended with a final cost of R11 507.21. while this is far above the target cost it is still far below the cost of many of the other rescue robots used in disaster situations. Trying to keep the robot low cost had various effects including the robot being underpowered.

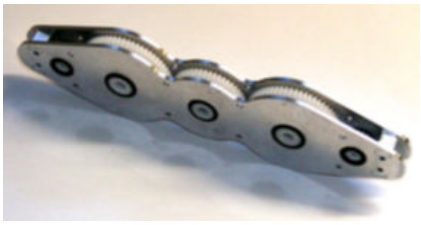


Figure 2.3: Load Intuitive Module.

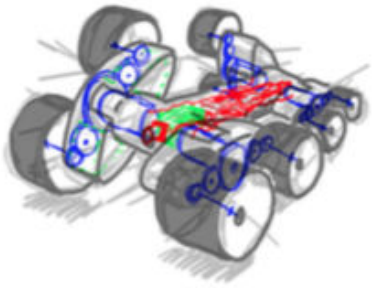


Figure 2.4: Concept sketch by Matthew Wilson.

2.4 Mechanics

2.4.1 LIM Design

The design of Load Intuitive Modules (LIMs) was done in the previously mentioned thesis by Matthew Wilson [22]. The design is based off the basic operation of an epicyclic gear box and can be treated similarly for calculations. The method of tables can be used to analyse an epicyclic gearbox. The method of tables is helpful in analysing how each component will move (including the frame) dependent on which gear is fixed, this will be vital to analyse the LIM system and calculate a gear ratio.

2.4.2 Gears vs Pulleys

Two methods for transferring mechanical power to a load are the use of gears and pulleys. LIMs could be achievable using either as long as slipping does not occur. Gears are better suited to high torque applications as they are less prone to slip, however pulleys are much cheaper and simpler to manufacture.

2.4.3 Motors

For the application of small scale electronics DC motors are commonly used. There are various kind of DC motors. Brushed motors are cheaper than brushless motors but brushed motors are less efficient than brushless motors. Small scale DC motors will be considered. Series wound DC motors can be easily modeled and the equivalent circuit can be seen in figure 2.5.
The operation of a series wound DC motor can be described using equations derived from this model.

$$e_a = V - i_a * R_a \tag{2.1}$$

From these equations and the fact that the speed of DC motor is proportional to the back EMF e_a equation 2.2 can be found.

$$\omega_m \propto \frac{e_a}{\phi_f} \propto \frac{(V - I_a * R_a)}{\phi_f} \tag{2.2}$$

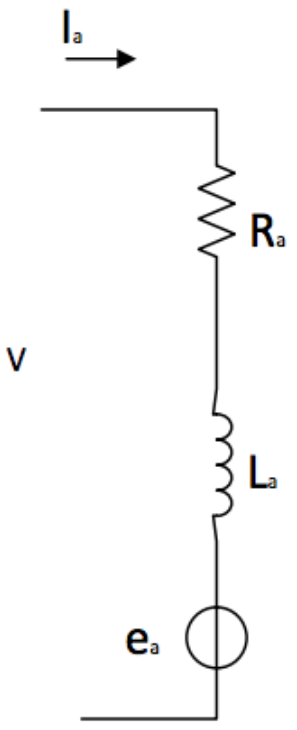


Figure 2.5: DC motor equivalent circuit.

From equation 2.2 it can be seen that changing the speed of the DC motor can be achieved by:

- Armature voltage control
  This is when field current is kept constant and armature voltage is varied.

Speed then changes according to :

$$\omega_m \ \alpha \ (V - I_a * R_a) \tag{2.3}$$

- Field control This is when armature voltage is kept constant and field current is varied.
  Speed then changes according to :

$$\omega_m \ \alpha \ \frac{e_a}{\phi_f} \tag{2.4}$$

$$\omega_m \ \alpha \ \frac{(V - I_a * R_a)}{\phi_f} \tag{2.5}$$

2.4.4 Torque Requirement

The Torque requirement of the motor and gearbox selection will depend on the weight of the body and the weight and length of the LIM-wheel assembly. The mass of the wheel will be calculated using

$$M = \rho V \tag{2.6}$$

Where for a cylinder such as a wheel

$$V = \pi r^2 h \tag{2.7}$$

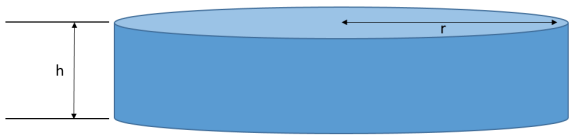


Figure 2.6: Dimensions for calculation of volume of a cylinder

By using eq. 2.6 and eq. 2.7

$$M = \rho \pi r^2 h \tag{2.8}$$

The weight of any parts that have pieces cut out to reduce weight can be calculated in the same way by using the formula above and then subtracting the weight of any parts that are cut out. Solidworks uses this formula to calculate the weight of any parts designed in it, this would be suitable for more complicated parts that have many various sized cut outs.

By taking the moments of these masses about the stationary point of the rotation of the assembly, the required torque to rotate the assembly can be calculated with eq. 2.9

$$T = (M_1 * d_1) + (M_2 * d_2) + (M_3 * d_3)... \tag{2.9}$$

Where M_i is the mass of the i^{th} object and d_i is the length of its moment arm.

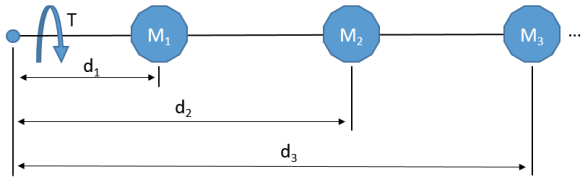


Figure 2.7: Diagram for calculation of moments

2.4.5 Materials

The trade off between materials must be carefully considered. In some cases a compromise will need to be made as using cheap materials that lack quality in some aspect will not satisfy the design requirement. If the material is too weak the structural integrity will suffer. If a material is too heavy a more powerful motor will be required which will increase the cost of the motor. In order to reduce weight and cost, the design must use as little material as possible. For example cutting holes into parts that are laser cut will reduce weight but will not directly reduce the cost.

Table 2.1: Material Properties

Material	Density [kg/m^3]	price	Shaping Method
Wood (hardboard)	800–1040	Low	Laser cut (easy)
Perspex	1180	Medium	Laser cut (moderate)
Foam	30-200	Low	Laser cut (difficult)
Plastic	1050-1200	High	3D printed

The Foam discussed is "proprietary expanded polyethylene foam manufactured by Sondor Performance Foams and sold in varying densities" as in the scarab project [21]

2.5 Electronics

2.5.1 Accelerometer

In order to determine the orientation of a body accelerometers are often used. There are various physical phenomena that allow an accelerometer to achieve a measurement of acceleration. Dave Redell [23] explains that the best way to understand how an accelerometer works and correctly utilize it is to forget about the inner workings of the accelerometer and rather analyse its black box performance. This can be taken with a pinch of salt as it is important to know how an accelerometer works and understand why you are getting certain data out of it. However, this was an indication that the type of accelerometer used would not be relevant as long as it performs the intended function. He goes on to explain that "An accelerometer is a device that senses deviation from free fall." This means that if the accelerometer is in free fall its reading will be zero and if the accelerometer is stationary it will read the acceleration by which it is deviating from free fall i.e. 1g. Using this along with the knowledge that the robot will not have vertical acceleration the strength and direction of the gravitational field acting on it can be deduced as being equal and opposite to the accelerometer reading. This knowledge will aid in being able to flip the camera feed of the robot in a situation where it is flipped upside down. This will allow for operation of the robot to be far more intuitive.

2.5.2 H-Bridge

"An H bridge is an electronic circuit that enables a voltage to be applied across a load in either direction. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backwards."[25]. H-Bridges are a common tool used with the operation of motors and an H-Bridge poses a solution to the issue of a rescue robot being able to apply full battery voltage to the motors in a forward or reverse direction at will. The circuit consists of 4 switches (these switches could be implemented using Bipolar Junction Transistors (BJTs), Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) or even mechanical switches) which allow for the selective flow of the full voltage through the motor. The name H-Bridge comes from the layout of these four switches which can be seen in figure 2.8.

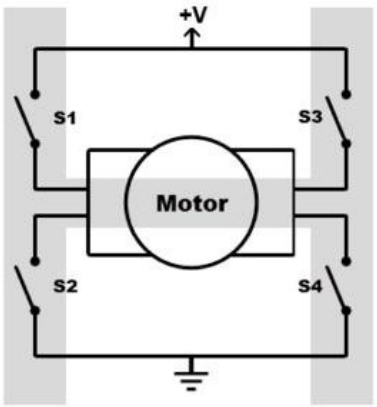


Figure 2.8: Basic schematic of an H-Bridge [24].

Shoot Through

whilst using an H-Bridge shoot through is extremely undesirable and occurs in the case where two switches on one side of the H-bridge are closed at the same time (i.e switch 1 and 2 or switch 3 and 4). This condition effectively creates a path from the high

voltage of the circuit directly to ground through the two transistors that when closed act the same as a piece of wire. This results in large currents which could damage the components in the H-Bridge or other components in the system. For this reason as well as the consideration of the huge amount of battery charge lost (if the components do not get damaged) shoot through must be avoided at all costs.
The truth table explaining the operation of the H-bridge shown in figure 2.8 can be seen in table 2.2

Table 2.2: H-Bridge operation truth table
*1 indicates closed switch
*0 indicates open switch

S1	S2	S3	S4	Effect
1	0	0	1	Motor Runs Forward
0	1	1	0	Motor Runs Backward
0	1	0	1	Motor Brakes
1	0	1	0	Motor Brakes
0	0	1	1	Shoot Through
0	1	1	1	Shoot Through
1	0	1	1	Shoot Through
1	1	0	0	Shoot Through
1	1	0	1	Shoot Through
1	1	1	0	Shoot Through
1	1	1	1	Shoot Through
0	0	0	0	None
0	0	0	1	None
0	0	1	0	None
0	1	0	0	None
1	0	0	0	None

Speed Control
H-Bridges can be operated with speed control by using PWM signals at the switches. To control the speed of a motor a PWM signal can be used to switch the transistors to alternate between having a voltage across the motor and no voltage across the motor, this creates an effective average voltage [26]. The duty cycle of the PWM can be varied to vary the average voltage. Increasing the duty cycle will increase the average voltage and decreasing the duty cycle will decrease the average voltage. This changes the speed through armature voltage control.

2.5.3 Remote Control

Radio Controlled (RC) remote and receiver pairs use various communication protocols to interface with. Owing to a standardisation of hobbyist RC devices most RC transmitter, receiver pairs communicate with servo motors and as such the most common of these protocols are Pulse Width Modulation (PWM) and pulse position modulation (PPM). Some RC receivers also make use of a satellite receiver which is connected to the main receiver by three wires; power, ground and signal. The signal line allows the two devices to communicate. If these satellite devices communicate with the main receiver using PPM a satellite receiver could also be used for the desired purposes. Information pertaining to the interface protocol of satellite receivers does not seem readily available online. This limitation of literature is not significant as information pertaining to main receivers that make use of ppm is readily available.

PWM-Pulse Width Modulation

PWM communication takes the form of an analogue pulse. The pulse has a specific period so that the receiver can tell when the pulse begins and ends. The length of the pulse represents a value that is being transmitted/received. The length of the pulse also falls within a standardised range. The values of the period and range of pulse length vary from application to application but in general the period is 20ms and the pulse length is 1-2 ms. An example of this can be seen in figure 2.9

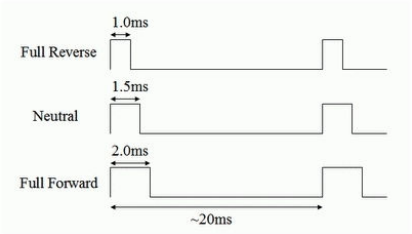


Figure 2.9: PWM structure example [27]

RC transmitter receiver pairs can vary in the number of channels that they can transmit and receive. The OrangeRx R617XL is capable of receiving 6 channels. Each channel will need its own; signal, power and ground lines as can be seen in figure 2.10. Having all these lines for each channel is convenient if the receiver is being plugged directly into multiple servos as is the case with hobbyist applications, but it can become cumbersome for other applications such as driving an H-Bridge from a micro controller which is receiving information from an RC controller as is the case in this project.

Figure 2.10: Servo connections [28]

PPM-Pulse Position Modulation

PPM communication works in a similar fashion to PWM however there are multiple pulses stacked onto one signal line. Each channel has a corresponding pulse that occurs at a specific instance in the chain. For example channel one's pulse will occur first

followed by a signal low for a specified period of time after which channel two's pulse will occur and so on. The PPM frame is generally 22.5ms the channel impulses range from 0.7-1.7ms and each channel has an end channel low signal of 0.3ms. The structure can be seen in figure 2.11. The result of PPM is that information from muliple channels is sent in one signal line.

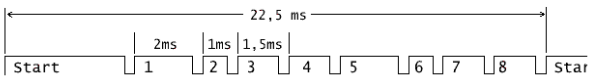


Figure 2.11: Structure of a PPM signal [29]

Control Layout

Other than the technical operation of the remote control, previous rescue robots show remote control configurations that are not intuitive. Complex controller layouts lead to necessary training and practice hours in order to have an operator effectively control the robot and be successful in maneuvering difficult terrains. An example of an overly complicated controller layout from the undergraduate mechanical project discussed in section 2.3.3 can be seen in figure 2.12. The controls for the rescue robot in question should be simple and intuitive to use. However the controls should not become so simple as to limit the maneuverability of the robot.

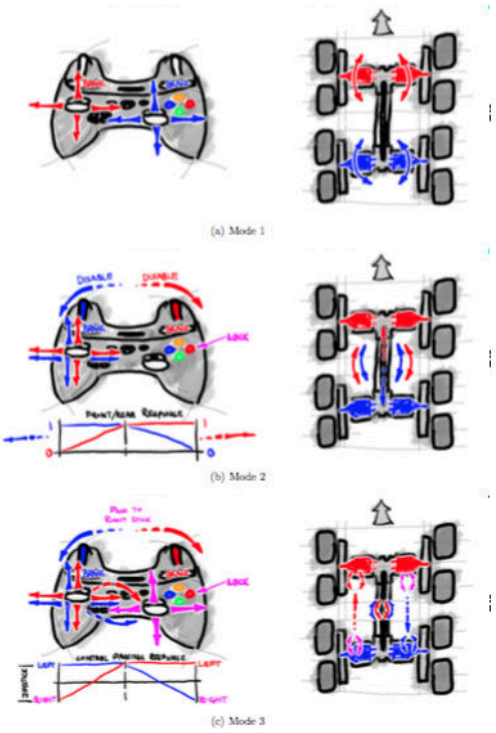


Figure 2.12: Controls from Matthew Wilson's project [22]

2.5.4 Microcontroller

In order to integrate the multiple electronic components of the project a microcontroller will need to be used. Owing to the small number of sensors to interface with and the limited applications for the microcontroller to complete, the only requirements are that the microcontroller has either SPI or I2C capabilities and can create 2 separate PWM signals. If more sensors were to be added and the project were to turn towards autonomy in a future iteration it is possible that a more comprehensive microcontroller would need to be used. Based on past experience with the hardware the STM32F051C6 will be used. This microcontroller forms a part of the microcontroller part of the education program for mechatronics students at UCT, as such a pre-manufactured development board is available to be used in order debug the code and run modules of the final design on their own, before they are integrated into the full working system

2.5.5 Batteries

For the robot to be able to have a long range it will need to be cordless which requires on-board batteries. Selecting batteries will need to be an exercise in balancing cost, weight and capacity. Disposable batteries would be wasteful as no matter how much is left in batteries from a previous run they would need to be swapped out for new ones to maximize time in the field. Rechargeable (secondary) batteries would be more appropriate. Rechargeable batteries can be bought in pre-made configurations or custom made. Selecting the appropriate battery chemistry and configuration can help balance these specifications. Some battery information gathered can be found in table 2.3

Table 2.3: Battery information

Chemistry	Energy Density [Wh/kg]	Typical Cost USD/volt	USD/Cycle
NiCd	45-80	6.94	0.04
NiMH	60-120	8.33	0.12
Li-ion	110-160	13.89	0.14
Lead Acid	30-50	4.17	0.10
Li-Poly	100-130	13.89	0.29

2.5.6 Video Feed

In order for the robot to be effectively driven the operator will need to see live video feed from the robot. This will allow the operator to see where the robot is going and if there are any survivors. There are a few options in order to achieve this. One relatively simple option is to use a small camera with a First Person View (FPV) transmitter which will link to a monitor or goggles with FPV receiver. This solution seems simple and effective as long as a strong enough transmitter and receiver are used. This solution is also cost effective with the camera and transmitter costing around 304 ZAR, this price includes

a camera and FPV transmitter. The cost of the receiver and display are less significant because they are out of the way of any harm and would not need to be considered written off if the robot were lost in a disaster zone. There are two options of transmission in FPV. The two options can be seen in table 2.4

Table 2.4: FPV camera encoding systems

Types	Frame size	Frame rates [fps]	aspect ratio
Phase Alternating Line (PAL)	720 x 576	25	4:3
National Television System Committee (NTSC)	720 x 486	29.97	4:3

Either of these encoding systems can be used with most hardware and neither have significant differences that would affect the project at hand.

3 Specifications

In order to have an informed and goal oriented design process specifications must be set out beforehand. These specifications will act as a guideline for the design process by clearly stating what the project should achieve. These specifications are based on the needs expressed in the literature review and aim to solve the issue at hand.

3.1 Size

The robot has the objective of navigating as many different environments as possible, for this reason the robot must be able to fit into small spaces. The robot also needs to be easily transportable. Keeping this in mind the size requirement for the robot was chosen to be within the following dimensions:

- Height: 160mm
- Width: 300mm
- Length: 400mm

3.2 Agility

The robot will encounter multiple forms of obstacles. The robot will be required to climb up a standard step sized object. This means the robot will need to be able to overcome an objects between 103mm and 178mm high. However, it was found in practice that mosts steps are around 145mm high. The robot will also be required to operate with no specified right side up, that being if the robot is flipped upside down from how it started it should be able to be operated as if nothing has changed. This requires inversion of controls and inversion of camera feed.

3.3 Cost

The robot is intended to be a cost effective robot that if lost needs to be considered disposable. In order to meet this constraint it is decided that the robot must cost under R4000. This would allow the fire rescue departments mentioned in section 2.3.1 to purchase multiple robots and not have to worry about them being occasionally damaged or lost in the field.

3.4 Vision

In order for the operator to be able to drive the robot and find survivors, the robot must have a camera with a live feed to the operator. This camera must have a resolution high enough to see survivors and must have a light for dark environments. For the resolution to be good enough to see obstacles and survivors 720x486 was deemed suitable.

3.5 Tele-operation

In order to deal with a high variety of disaster situations from a safe operating distance the robot should be able to be operated without the need for a physical tether. Operation should be possible from a range of 300m.

3.6 Battery Life

The robot is required to enter a disaster situation complete its task and then return from the disaster zone to the operator. A short battery charge time will reduce costs as batteries are a significant portion of the budget. With a short charge time only two batteries will be required, one battery can be charged while the other is used and then the robot can return to the operator the batteries can be swapped and the other battery can then be charged and so on. This effectively results in no down time where the robot must sit idle while the battery charges. A short charge time will also allow the robot to be deployed quicker if the batteries have not been kept at full charge.
Keeping this in mind the robot must meet the following specifications:

- Battery usage time: 1 hour
- Battery recharge time: 1 hour

3.7 Robustness

In order for the robot to be robust it must be able to withstand driving into a step sized object at full speed as this is the most realistic metric for what the robot would encounter other that debris collapsing on it. A further robustness specification is that the motors should not stall while climbing a step and the wheels and gears should not come loose.

3.8 Simplicity of Operation

One of the greatest issues in rescue robotics is a lack of personnel to operate the equipment, this is partially solved by the low cost of the robot as someone less skilled can be entrusted with a cheaper robot. In order to meet this specification the operation of the robot must be intuitive enough to be operated after very little practice.

4 Methodology

This section details the steps and processes followed in the design and implementation phases of the project. The processes followed were chosen to ensure that the objectives were met. Some of the processes followed had the intention of completing a task the first time around and others were planned to be iterative in order to gain practical knowledge in the process.

4.1 Design, Calculation and Simulation

The nature of the system required both a mechanical and electronic design. As a result of the large differences in the design of mechanical and electronic systems, vastly different approaches were used for each.

4.1.1 Mechanical Design Methodology

In order to achieve the goals required for the mechanical design an iterative; design, build, test and evaluate strategy was used as seen in figure 4.1. This was owing to a lack of prior experience with mechanical designs and the inherent nature of mechanical designs that allows problems to be easily seen through observation and physical testing.

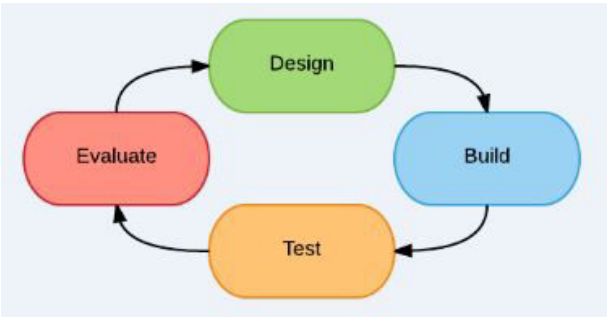


Figure 4.1: Iterative design procedure

Prototyping
In order to prevent prototyping from becoming a time consuming and costly exercise cost effective materials were used where necessary and low cost test materials were used for certain prototypes with the intention of confirming their accuracy before building

with the final material. It was also necessary to attempt to fix all current issues with a prototype in the design of the subsequent prototype in order to prevent iterations that yield no improvement.

Solidworks
Solidworks was used for the design and specific simulation of the mechanical build. This allowed confirmation of the fit of pieces, meshing of gears and correct operation of mechanisms.

4.1.2 Electronic Design Methodology

In order to achieve the goals required for the electronic design a more calculated and planned approached was followed. The process involved calculations and written design of circuits followed by testing the circuits on a breadboard to verify the correct operation of the circuit. This method was followed rather than an iterative process as experience in the field of electronics was not a major concern and when there are design problems in a circuit it is not as easily seen by the eye or through testing.

Logisim
In order to simulate certain logic circuits before implementation Logisim was used. This software allows for simple construction of circuits that can then be simulated with the program. The program shows lines that have a logic high in bright green and lines that have logic lows in dark green, this allows for easy debugging and verification of logic circuits.

Veroboard Design
Design of veroboard implementation was done by modifying an excel spreadsheet to look like veroboard and designing by editing the colours of cells to mimic wires.

4.2 Fabrication, Manufacture and Implementation

4.2.1 Mechanical Implementation Methodology

Fabrication, manufacture and implementation of the mechanical components formed a part of the design process owing to the iterative approach used.

Laser Cutting
A laser cutter was used to fabricate the parts designed, this was achieved through

exporting a PDF of the designed parts from Solidworks and uploading it to the laser cutter which performs the cutting withing a 0.2mm tolerance. The use of the laser cutter required all pieces to be designed as flat shapes to be cut and joined. This is owing to the fact that laser cutters cannot shape objects in three dimensions but rather only in two dimensions.

Construction
Where possible joining the fabricated pieces together was done in a way that allowed for disassembly when necessary in case anything broke or needed to be replaced owing to design changes. Cable ties, threaded rods, nuts and elastics allowed for an easy to disassemble configuration.

4.2.2 Electronic Implementation Methodology

Circuits
The implementation of electronics was completed through manufacture of circuits on veroboard. It can also be noted that design and manufacture for PCB would be advantageous to size of the circuits and the complexity and time taken to manufacture the circuits in large quantities.

Atollic
Modules that interface with the microcontroller were tested with the microcontroller in the UCT development board connected to a PC running Atollic. Atollic allows code written in C to be compiled and loaded onto the microcontroller. Atollic allows for tracking of variable values which is helpful when debugging the code and investigating the form of the information being received.

5 Design Planning

The design of the robot is a long spanning part of the project as such design philosophies had to be chosen and held by throughout the process in order to create a consistent and sensible design. The design philosophies also need to be given an order of importance so that more important concerns would take preference in decisions. The structure of the preference hierarchy can be seen in figure 5.1. The hierarchy is structured graphically in a pyramid shape with the most important aspects at the top of the pyramid and the least important aspects at the bottom.

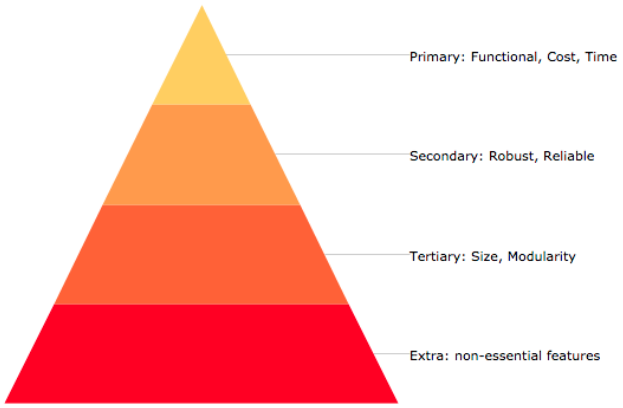


Figure 5.1: Design philosophy hierarchy

The fundamental design philosophies were:

- Keeping the robot cost effective
- Ensuring that the robot fulfilled its function
- making sure the robot was deployable in time

Neglecting any of these considerations would result in a final product that did not meet the criteria of building a cost effective search and rescue robot. Thus in order to make sure the robot operates as desired certain expenses would need to be incurred and these expenses would need to be justified by the fact that they were necessary in making the project work.
The secondary tier of design philosophies included ensuring a robust system was created as it is of little use if the system works and then ceases to work suddenly afterwards.

The tertiary tier in the design philosophy framework includes keeping the system within size specifications as well as designing for modularity allowing for the possibility of easy expansion. These goals could be seen as a bonus for the design as they offer improvements on the fundamental system which must work before these are pursued. The extras tier in the design philosophy is additional improvements not concerning the basic functions of the robot. These small improvements may improve the overall system, however all the considerations above these improvements must be fulfilled before these issues are pursued.

5.1 Function Oriented Design

The system performs multiple functions that must work at once, as such the system's functions cannot be designed in isolation and must all be considered during development. The microcontroller is responsible for controlling all functions except for the video feed and as such diagrams can be used to better express these functions and how they interrelate. In order to clarify the functions desired in the system a functional block diagram is used as shown in 5.2 . This diagram shows the sub-systems to be incorporated into the main system. A key is used to identify which systems are essential for testing and initial deployment and which systems are not. This will not only help clarify the functions to design and build but the order of importance that they have.

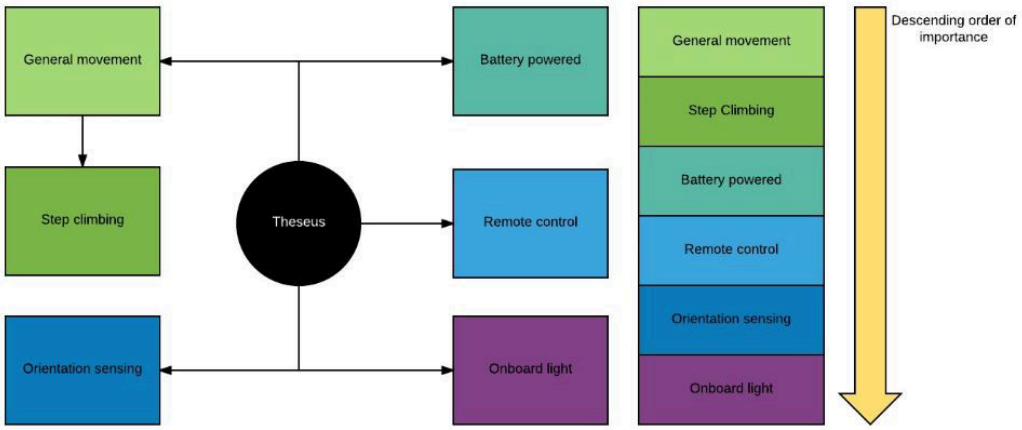


Figure 5.2: Functional block diagram

Once all the functions have been expressed in the basic functional block diagram, the functions can be further expanded with the aid of a more detailed sub-system drop down diagram. The detailed sub-system drop down diagram describes whether is is an electrical or mechanical design problem and what components will need to be used in order to fulfill the function (as well as the voltages they must be run off). This will allow for clarification on which systems are dependent on each other. This detailed sub-system drop down diagram can be seen in figure 5.3.

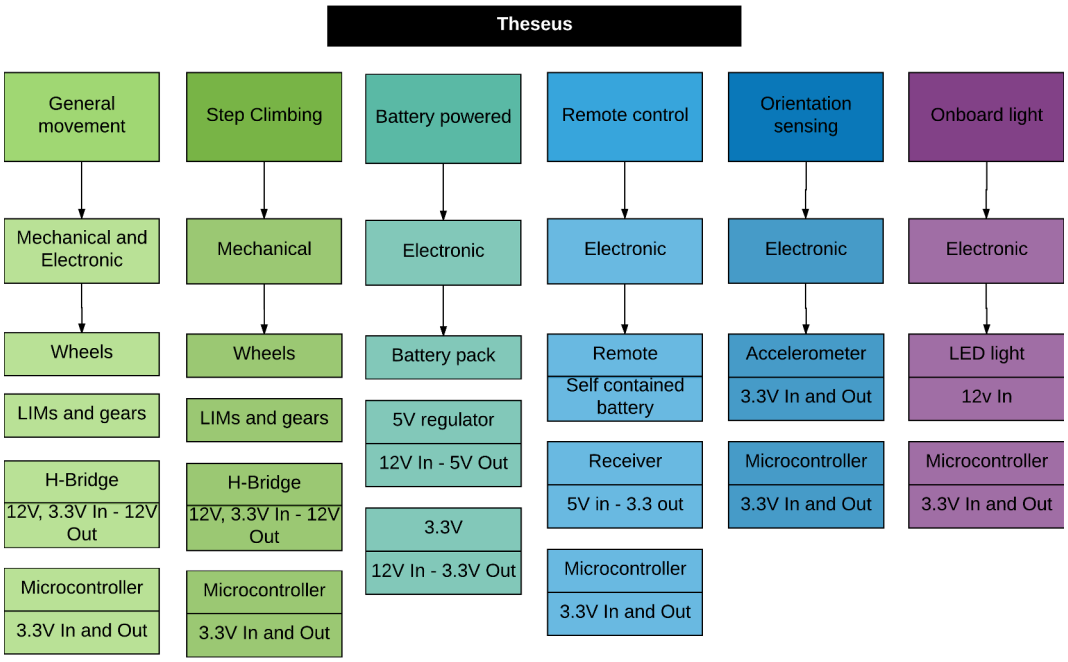


Figure 5.3: Sub-system drop down block diagram

Showing the interdependencies in the system helps streamline the designing and building phases as proper design planning can be implemented. Proper design planning will also reduce the chances of any critical sub-systems being left out owing to a lack of time, by prioritising critical sub-systems above non-critical sub-systems. Based on the diagrams presented in figure 5.2 and 5.3 all design and building will be arranged in order to complete any systems involved in driving and climbing followed by operating the system off a battery pack and setting up remote control communications and then finally concentrating on orientation sensing and the on board light. If delays occur in any of the critical sub-system processes that result in down time progress can be made on the next sub-system in the critical chain untill progress can be made on the "more critical" system again.

5.2 Milestone Oriented Design

In order to keep the project on track and directed towards the final goal a milestone based design procedure may be helpful. This will allow for an order of events to be created as well as a definition of what would constitute a finished sub-system or milestone. A flow diagram of important milestones and what needs to be considered in order to declare a milestone reached can be seen in figure 5.4. The milestone flow diagram is split into paths. Each path concerns a certain sub-section of interrelated design and building milestones. It must be noted that if a milestone becomes too difficult or time consuming to reach, a milestone independent of that one can be pursued until that milestone is again viable.

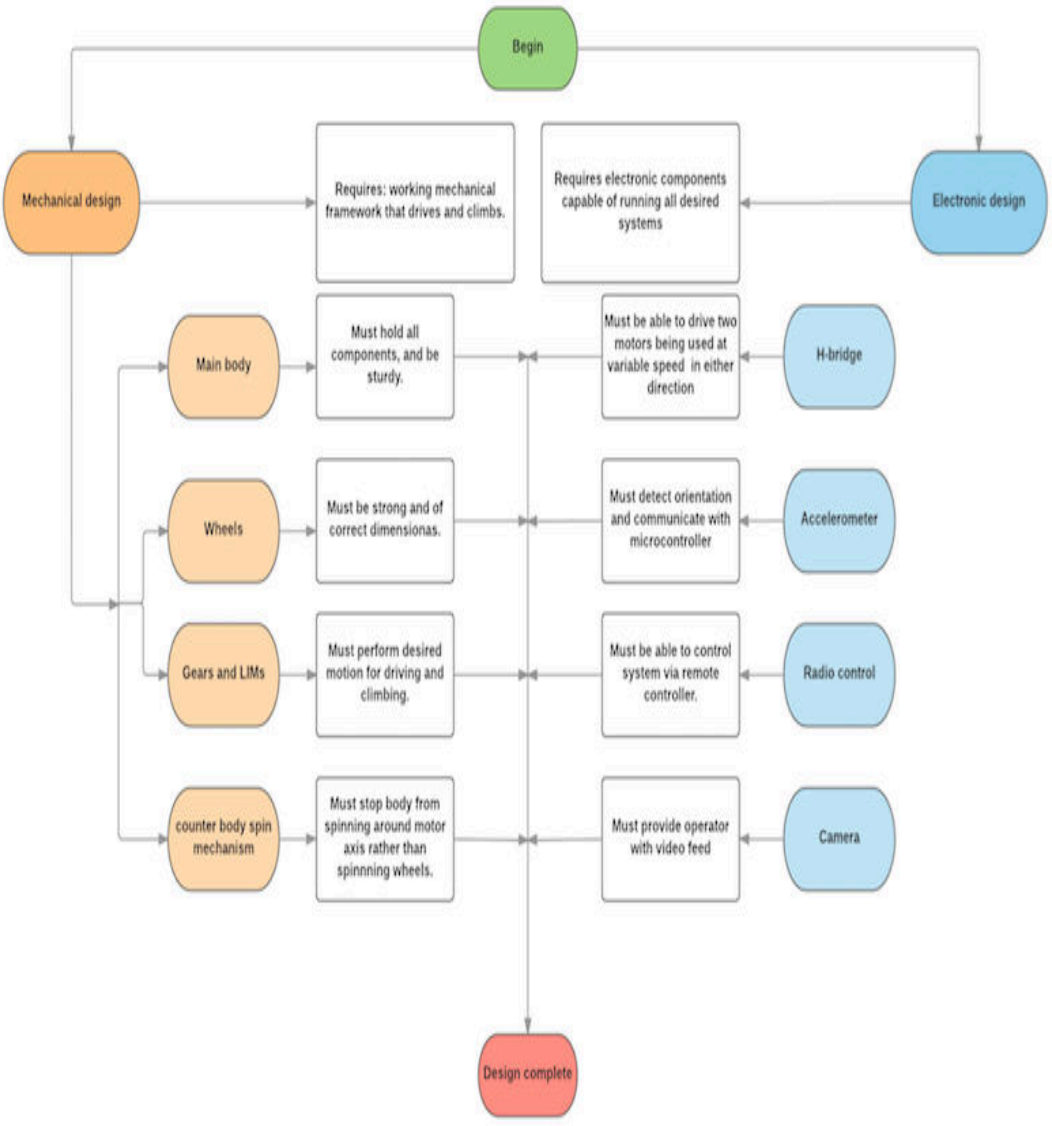


Figure 5.4: Simplified milestone flow diagram

The specifications in order to declare a milestone complete are vague but are worded in a way that if the milestones are adhered to in a strict manner and satisfied the final system will work. Work beyond the milestone requirements can be done with excess time if desired.

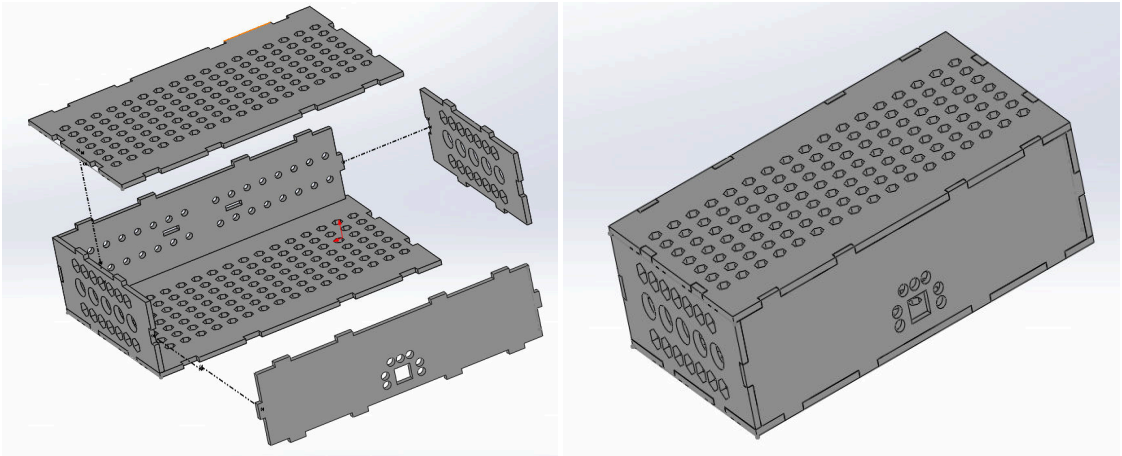
6 Design

This section will deal with the design process that went into creating each component necessary for the final system. The approach to design and manufacture included simulations and calculations as well as an iterative improvement approach. As such the sub sections in this design description will begin with the simulations, calculations and planning for the design followed by an explanation of the actual manufacture/build process and how the component was refined.

6.1 Mechanical Design

6.1.1 Body

The body was treated as the base of the system and as such was designed first. The body is designed to house all of the electronics and to have the gears and wheels externally attached to the motor shafts that protrude from the body. As everything was to be designed around the body the body was designed first.
The body is designed with rectangular pieces jutting out of the edges of some parts and rectangular gaps in others to allow the body to be laser cut into flat pieces that fit tightly together like a puzzle. These pieces form a box which can be glued in place using the rectangular cut outs. The final design of the main body can be seen in figure 6.1. Figure 6.1a shows an exploded view of the assembly which shows how the pieces can be joined together and figure 6.1b shows the assembled main body.



(a) Final main body design exploded view (b) Final main body design joined view

Figure 6.1: Final main body design

The box was designed so that one of the two largest sides would not be glued in place but would rather have a hinge on one end and a magnet on the other allowing one side of the box to be easily opened and closed. Access to the inside of the box is necessary to gain access to the internal circuits and motors.
The body schematics can be seen in appendix C. The body was designed to be large enough so as to fit the motors, circuitry and camera antenna whilst being as small as possible to avoid beaching the robot on obstacles which is more likely if the body of the robot is larger. The design proved to be robust with the joints offering a tight guided fit and a large enough surface area for adhesives.
The choice to include holes is owing to the desire to reduce weight as it was designed before the motor choice was finalised. Once the motor choice was finalised it became apparent that there should be enough torque produced and that the a reduction in weight would not be necessary. The holes also provide a small amount of cooling in the event that the motors begin to heat up.

The body was fabricated using hardboard and assembled as seen in figure 6.2. Once assembled the hardboard was deemed to be a suitable material as the box was sturdy and seemed to be capable of withstanding any reasonable forces applied to to it. The disadvantages of the way the body is designed are a concern in wet or damp environments. These disadvantages are the fact that wood even though treated with paint may degrade after contact with water and the holes expose the circuitry to any water that may splash up or fall on the robot. It was also considered that the weight saved would not be justified by the cost of the extra time used by the laser cutter. These disadvantages were not deemed to be serious enough for an immediate redesign however they should be considered if another design were to take place.

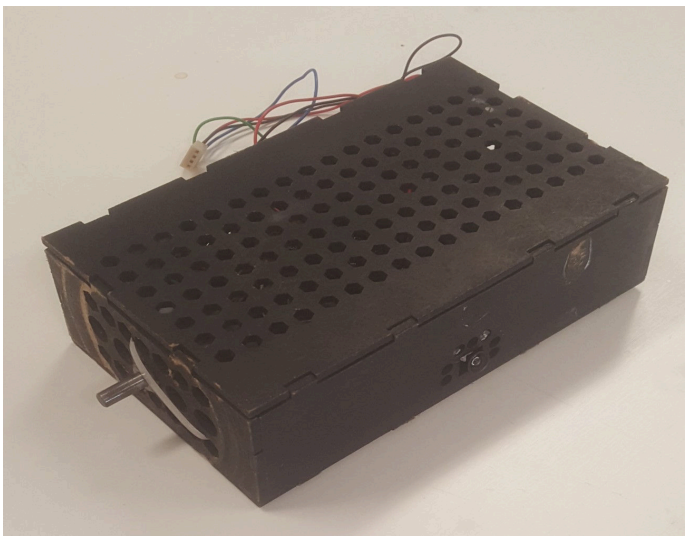
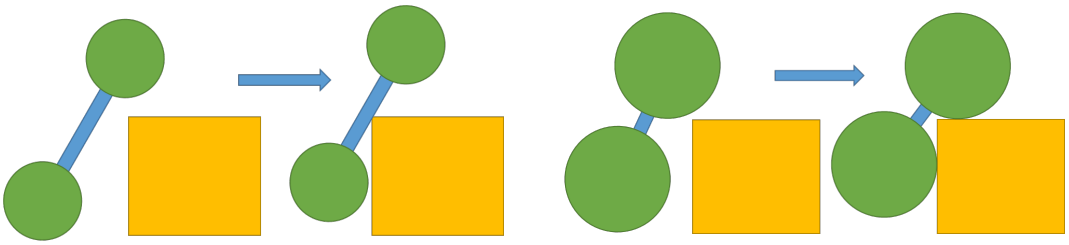


Figure 6.2: Fabricated and assembled body containing components

6.1.2 Wheels

6.1.2.1 Size

In general a wheel can only overcome obstacles that are the size of the radius of the wheel or smaller. Therefore in order to overcome a standard step the wheel would need to be significantly larger than the step this begins to become a problem for robots that need to fit into tight spaces as the large wheels will become too big to fit. In order to remedy this issue the LIM system is effective. The LIM system is effective as it allows a wheeled robot to overcome obstacles that are double the size of the wheel or bigger whereas a traditional wheel design only allows the robot to overcome obstacles around half the size of the wheel or smaller. This allows for the use of wheels that are half the size of what should be needed (allowing it to fit in tight spaces) without affecting the climbing ability of the robot. A wheel size must be determined to achieve the goal of climbing a step. A wheel with diameter 125mm was chosen as this would allow the second wheel to flip over onto the step and gain traction with ease while still being smaller than the largest sized standard step. The wheels could be made smaller with longer LIMs however there is the problem of beaching the robot. Beaching refers to the case when the LIM flips, the LIM bracket lands on the edge of the step and the wheel cannot gain traction on the edge or side of the step. An example of how the size of the wheels and the distance between them can effect beaching can be seen in figure 6.3. Figure 6.3a indicates how small wheels and a large space between them results in beaching and figure 6.3b indicates how larger wheels with a shorter gap between them can help avoid beaching. It can be seen in both figures that the same size obstacle is being used but by changing the wheel size and distance between wheels the likelihood of beaching can be changed.



(a) Effect of small wheels with large gap on beaching.

(b) Effect of large wheels with small gap on beaching.

Figure 6.3: Simplified diagrams to indicate the effect of the size of wheels and gaps on beaching

6.1.2.2 Material

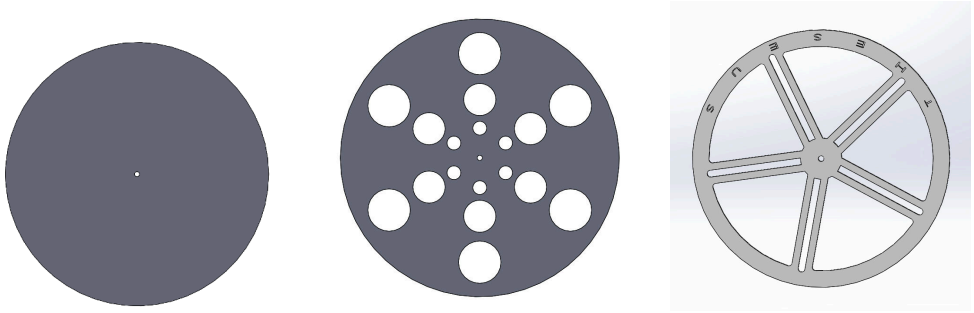
The material of the wheels was decided based upon table 2.1 which left hardboard and perspex under consideration owing to their ease of fabrication using a laser-cutter as well as their cost effectiveness. Hardboard was chosen for the initial design as it was cheaper, lighter and is less brittle than perspex.

6.1.2.3 Weight

The weight of the wheels was reduced as much as possible before the motor selection occurred as the wheel are a large contributor to the torque requirement from the motor. Reducing the weight of the wheels was done by choosing an effective material as discussed in subsection 6.1.2.2 as well by implementing cut outs and keeping the wheels narrow.

Cut Outs

Shapes where cut out of the wheel where possible in order to reduce the volume of the wheel and hence the weight. Several designs were completed, these can be seen in figure 6.4 and a final design was chosen for its balance between strength and low weight which is shown in figure 6.4c



(a) Solid wheel design

(b) Patterned hole wheel design

(c) Starfish patterned wheel design

Figure 6.4: Wheel pattern designs

The patterned hole design in figure 6.4b provides a wheel that is only 66% of the weight of the solid wheel in figure 6.4a. The Patterned "starfish" design in figure 6.4c provides a wheel that is only 32% of the weight of the solid wheel in figure 6.4a.

Narrowness

Narrow wheels were used as this reduces the volume and thus the weight. It must be noted that this would not decrease grip. Having wider wheels does not mean better grip. Grip is a term used to describe friction. Friction is the product of the normal force of the surface on the wheel and the coefficient of friction between the two surfaces as in equation 6.1

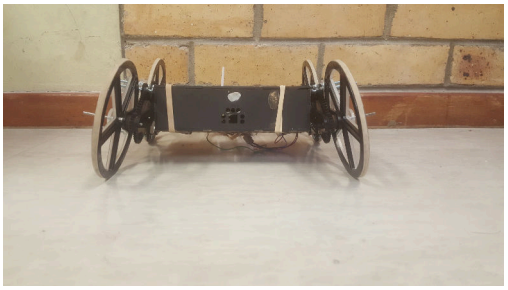
$$F_f = N * \mu \tag{6.1}$$

Neither the normal force nor the coefficient of friction change when the width of a wheel is changed. The reason that thicker tires are preferred (in areas such as Formula1 racing) is owing to the fact that then they experience less force per unit area and softer rubber can be used on the tire without ripping the material.

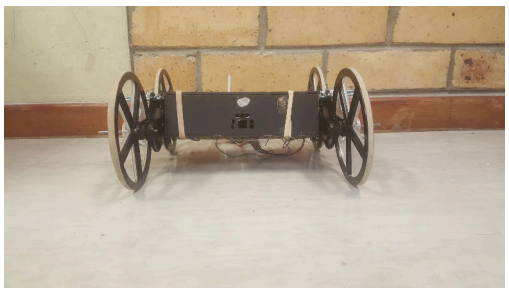
While the width is decreased in order to reduce weight a design consideration is that thicker wheels will provide more stability to the robot as there will be a larger surface area for the body to rest on allowing a better structural integrity. This is a concern as the entire weight of the body will be resting on the wheels which are only attached to the body through one shaft on each side.

Final Assembly

Once the final design was picked the wheels were fabricated by laser cutting them from 3mm hardboard. These wheels were used until the LIM design was complete after which it was found that the hardboard was warping and degrading rapidly causing loosening around the shaft ultimately resulting in a weak and unstable build. The hardboard wheels were replaced with 3mm perspex wheels this corrected the issues of degradation, warping and loosening however once assembled the wheels still leant inwards under the weight of the body which can be seen in figure 6.5a. The desired stability can be seen in figure 6.5b. This was remedied by stacking two wheels next to each other making them 6mm thick as well as adjusting the LIM design which will be discussed in section 6.1.5.3. This resulted in a sturdy and reliable design.



(a) Wheels buckling under the weight of the body



(b) Wheels holding the weight of the body

Figure 6.5: Examples of the wheels ability to hold the main body

6.1.3 Motor Selection

Various motors were researched and their specifications are given in table 6.1 these motors were compared as shown in table 6.1.

Table 6.1: Motor specifications

Motor	Torque [N.m]	price [ZAR]	Speed [RPM]	voltage [V]
1	0.147	103.89	400	12
2	0.147	79.05	1000	12
3	0.196	99.62	270	12
4	0.294	112.95	80	12

6.1.4 Torque Requirements

The torque required to flip the LIM assembly in the worst case scenario (flipping the LIM with the full weight of the body and all its contents resting on the center axle of the LIM) was calculated using the torque calculations described in section 2.4.4. The torque calculations were completed using estimates of weight from Solidworks and by weighing the electrical components to be used. These calculations were performed in Microsoft excel as seen in figure 6.6

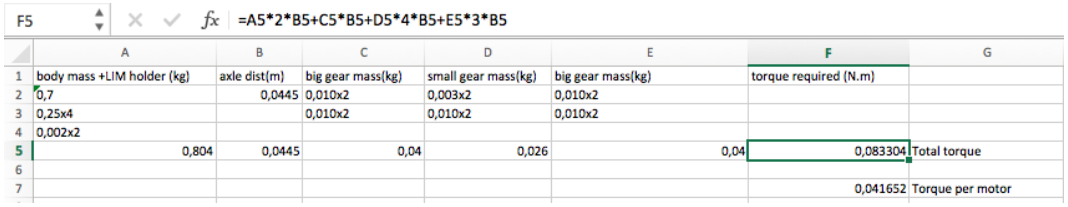


Figure 6.6: Calculation of rough torque estimates using Microsoft excel

These calculations resulted in an estimation of the need for each motor to produce 0.0417 N.m. to lift the whole robot in the event that the front wheel becomes stuck. From these rough estimates a table of the max torque reduction gear ratios that can be used for each motor before the torque produced will be too low to flip the LIM. These calculations have all been completed using worst case estimates and assumptions in order to have a safe margin of error.

Table 6.2: Torque requirements from motors

Motor	Torque required per motor [N.m]	Torque produced [N.m]	Max. gear ratio
1	0.0417	0.147	3.53
2	0.0417	0.147	3.53
3	0.0417	0.196	4.70
4	0.0417	0.294	7.10

Motor 3 was chosen as it produces enough torque and was readily available when the body design was being completed. This allowed the main body to be designed in order to fit the motors that were deemed to have sufficient characteristics. The cost of such motors was deemed an acceptable amount as operation of the robot would suffer if the robot was underpowered. The motor selected can be seen in figure 6.7. The dimensions can be seen in figure 6.8 as found on the data sheet.

Figure 6.7: Mantech EGB 12v (0.196N.m) motor

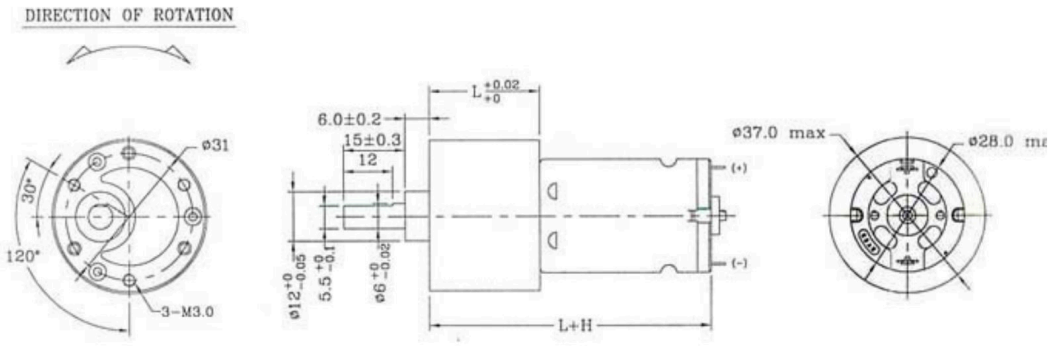


Figure 6.8: Motor schematic to indicate dimensions

6.1.5 LIMs

The LIM design was based on those in Matthew Wilson's undergraduate project [22] and then modified to be cost effective and larger to suit the design of Theseus.

6.1.5.1 Modularity

The LIMs are designed to be laser cut out of perspex or hardboard to be cost effective and lightweight. This also means that replacement parts are easy and quick to design and manufacture. The LIMs are designed to be easy to assemble and disassemble making them modular in the sense that any worn parts can be easily changed and that gears can be easily swapped to change gear ratios as deemed necessary. The LIMs attach and detach from the motor shafts protruding from the main body, this allows the whole LIM to be easily taken off and replaced when necessary.

6.1.5.2 Gear Ratio and Arm Analysis

The LIMs operate in a similar fashion to an epicyclic gearbox. This allows for analysis of the motion of the LIM by using a tabular method used in the analysis of epicyclic gearboxes. This analysis is derived from the analysis done in the undergraduate mechanical final year project in section 2.3.3. The labels in the analysis are based on those in figure 6.9 which serves as simple schematic of the kinematic model to graphically indicate how the gears have been labeled and identify a coordinate system.

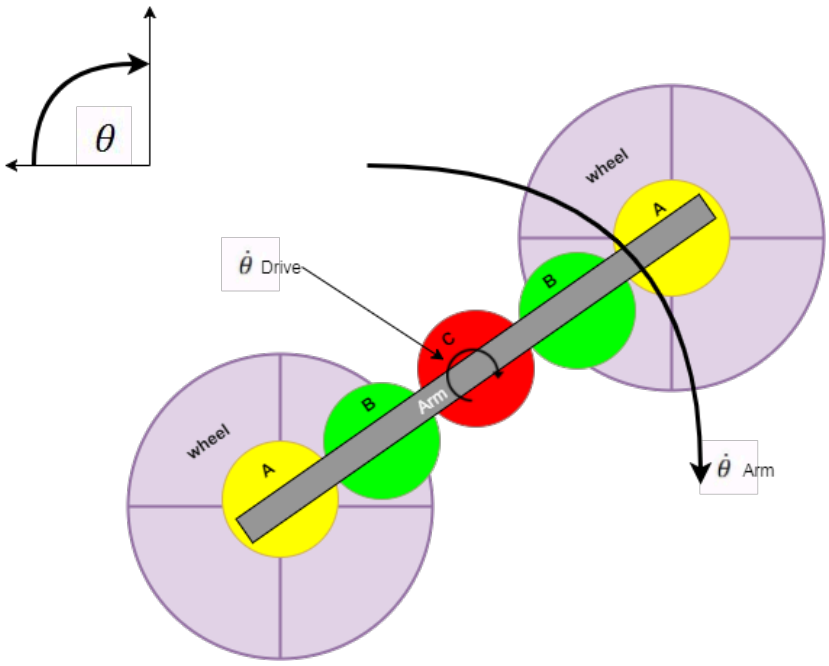


Figure 6.9: Simple schematic for gear labeling and co-ordinate definition.

In the calculations to follow generally accepted symbols are used, these symbols and what they represent are:

- θ - Represents the angular position of an object from the negative y-axis measured in degrees as defined in figure 6.9 on the previous page. The clockwise direction is taken as positive by the definition given in figure 6.9 on the previous page.
- $\dot{\theta}$ - Represents the rate of change of the angular position of an object in degrees per second. The clockwise direction is taken as positive by the definition given for θ

The gear analysis can be seen in table 6.3.

Table 6.3: Motion analysis of LIM

	A	B	C	Carrier Arm
motion relative to frame	$\dot{\theta}_{Wheel}$	$-\frac{N_A}{N_B}\dot{\theta}_{Wheel}$	$\frac{N_A}{N_C}\dot{\theta}_{Wheel}$	$\dot{\theta}_{Arm}$
motion relative to carrier arm	$\dot{\theta}_{Arm}$	$\dot{\theta}_{Arm}$	$\dot{\theta}_{Arm}$	0
Absolute motion	$\dot{\theta}_{wheel} + \dot{\theta}_{arm}$	$\dot{\theta}_{Arm} - \frac{N_A}{N_B}\dot{\theta}_{Wheel}$	$\dot{\theta}_{Arm} + \frac{N_A}{N_C}\dot{\theta}_{Wheel}$	$\dot{\theta}_{Arm}$

From this table analysis of the gear linkage is possible. Various conclusions can be drawn about the effect of the gear ratio that allows for analysis of the gear linkage:

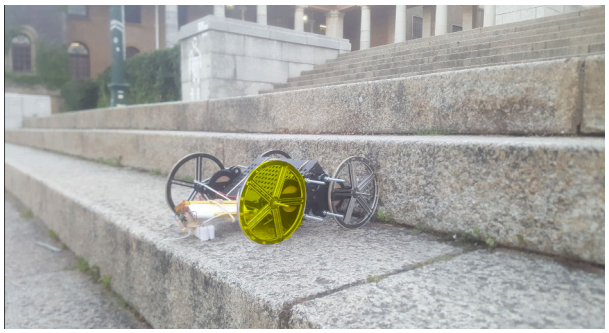
- The drive speed is equal to the absolute speed of gear C as the motor drives gear C directly, Thus:
$$\dot{\theta}_{Drive} = \dot{\theta}_{Arm} + \frac{N_A}{N_C}\dot{\theta}_{Wheel} \tag{6.2}$$
- If the carrier arm is not rotating (robot is driving forwards) $\dot{\theta}_{Arm} = 0$
$$\dot{\theta}_{Drive} = \frac{N_A}{N_C}\dot{\theta}_{Wheel} \tag{6.3}$$
- If the wheels are not moving (the wheel is stuck) $\dot{\theta}_{wheel} + \dot{\theta}_{arm} = 0$
$$\dot{\theta}_{Drive} = \dot{\theta}_{arm}(1 - \frac{N_A}{N_C}) \tag{6.4}$$
$$\dot{\theta}_{arm} = \frac{\dot{\theta}_{Drive}}{1 - \frac{N_A}{N_C}} \tag{6.5}$$

Equation 6.5 bears importance on the design and can guide the choice of gear ratio. This equation helps identify the effect that various gear ratios will have on the robot while it is driving and climbing. Various effects of the gear ratio can be analysed by looking at the denominator of the right side of equation 6.5, $(1 - \frac{N_A}{N_C})$. Once the gear ratio ($\frac{N_A}{N_C}$) has been substituted in it can be seen whether the resulting denominator (and therefore fraction) is positive or negative. The result being positive or negative will have a great

impact on the motion of the LIM as the sign will indicate the direction that the arm (LIM bracket) rotates. It can be seen that:

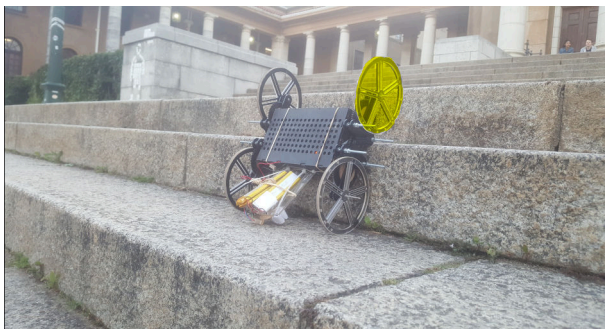
- If $N_A < N_C$:
  The arm rotates in the same direction as the drive and the wheels. This is the desired motion for climbing objects and steps through rotation of the back wheel on to an object when the front wheel becomes stuck
- If $N_A = N_C$:
  The drive is not capable of rotating the arm. No work can be done by the drive at this gear ratio.
- If $N_A > N_C$:
  The arm rotates opposite to the direction of the drive and wheels, this would help the robot overcome small objects with the front wheel but could cause complications with the back wheel overcoming the obstacle. This gear ratio would result in the body rotating in the same direction as the drive which could cause flipping of the body rather than the LIM if a wheel becomes stuck.

The desired motion is that stated in the case where $N_A < N_C$. In this case the back wheels will flip over onto obstacles in order to climb them. This motion can be seen in figure 6.10 on the following page. Figure 6.10 is made up of six figures that are presented in the order that the motion should occur, starting with figure 6.10a indicating the front wheels coming into contact with the obstacle. Figure 6.10b follows showing the rear wheels beginning to lift while the front wheels remains still, this is followed by the rear wheels rotating over the front wheels shown in figure 6.10c. Figure 6.10d then shows the rear wheels making contact with the obstacle and figure 6.10e shows the robot lifting itself up by using the rear wheels which are now on top of the obstacle. Figure 6.10f shows the completion of the motion by overcoming the obstacle. In these figures the wheels that were at the back at the start of the motion is highlighted throughout the motion to allow the progression of the motion to be easily followed.

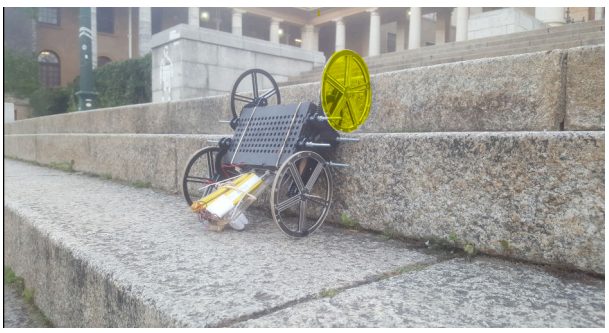


(a) step 1 of the desired motion for rotating the rear wheel on to the obstacle

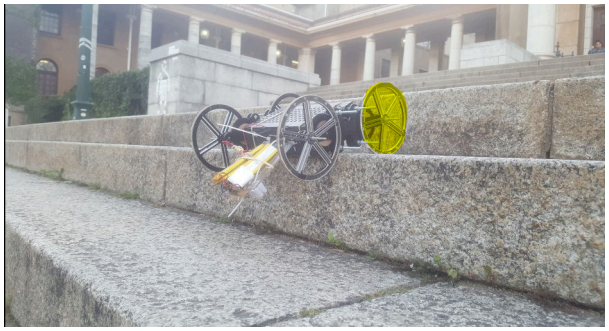
(b) step 2 of the desired motion for rotating the rear wheel on to the obstacle



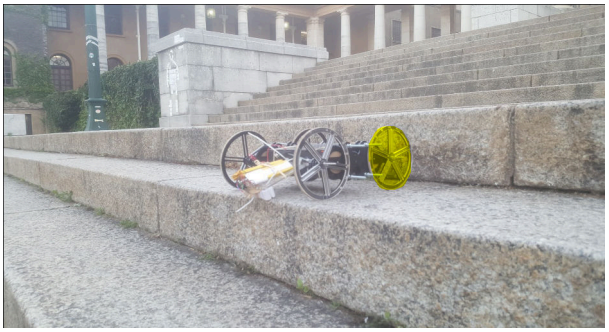
(c) step 3 of the desired motion for rotating the rear wheel on to the obstacle



(d) step 4 of the desired motion for rotating the rear wheel on to the obstacle



(e) step 5 of the desired motion for rotating the rear wheel on to the obstacle



(f) step 6 of the desired motion for rotating the rear wheel on to the obstacle

Figure 6.10: Demonstration of steps for the desired motion to rotate the rear wheel on to an obstacle

6.1.5.3 Iterative Improvement Procedure

LIM Version 1.0

A gear module of 1 was chosen as from testing with laser cutting it was found that this module provided well cut gears that mesh well, without making the gears overly large. The gear ratio was designed to span the necessary distances while providing enough torque/speed based on the calculations in section 6.1.4. In the initial design of the LIM The number of teeth on each gear is:

- 45 teeth on the center driver gear (paired to motor)
- 44 teeth on the idler gears
- 45 teeth on the driven gears (paired to wheel)

In order to ensure that these gears fit in the LIM bracket equation 6.6 can be used.

$$(Number\ of\ Teeth) * (Module) = Diameter \tag{6.6}$$

Using equation 6.6 and knowing that the distance between the central hole and the wheel hole on the LIM bracket is 89mm it can be confirmed that
$\frac{45}{2} + 44 + \frac{45}{2} = 89$ i.e. the distance between holes in the LIM bracket.

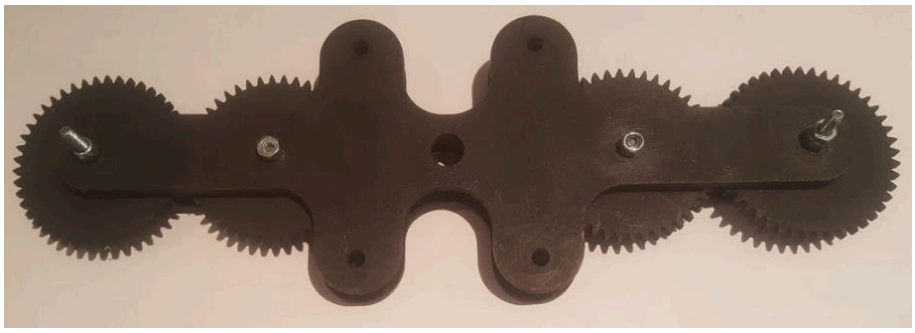


Figure 6.11: LIM version 1.0 fabricated from wood and assembled

These gears were laser cut from hardboard and assembled with the LIM bracket as seen in figure 6.11. The wheels have not been attached in figure 6.11 however the wheels are coupled to the two outermost gears by the long threaded rods depicted in the figure. The motor is coupled to the lim through the central hole in the bracket which feeds into the middle of the centre gear. Testing the meshing of the gears and the fit onto the motor shaft demonstrated shaking and misalignment of the gears as a result of going off centre on the axles and ill fitting on the shaft. The shaking worsened owing to the rapid degradation of the hardboard. It was determined that hardboard would not be suitable for the gears or the LIM brackets.

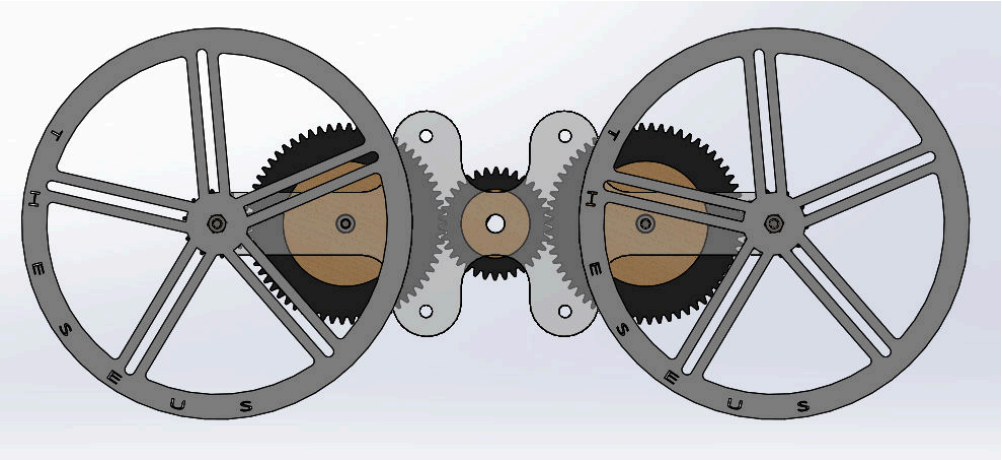
LIM Version 2.0 and 2.1

The gears and brackets were redesigned to be cut from perspex with a gear ratio based on the information in section 6.1.4. This design also took into account the design conclusions drawn in section 6.1.5.2. In this design the number of teeth on each gear is:

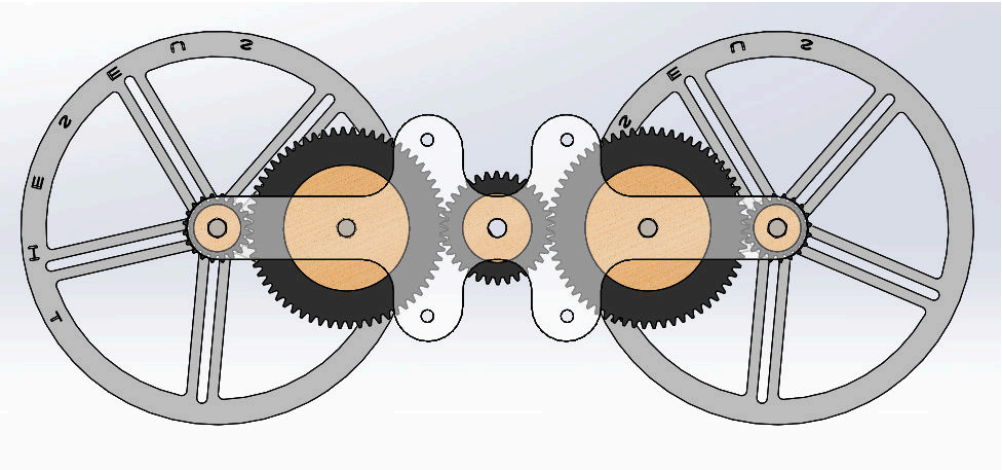
- 34 teeth on the center driver gear (paired to motor)
- 62 teeth on the idler gears
- 20 teeth on the driven gears (paired to wheel)

Using equation 6.6 and knowing that the distance between the central hole and the wheel hole on the LIM bracket is 89mm, it can be confirmed that: $\frac{34}{2} + 62 + \frac{20}{2} = 89$

These tooth numbers result in an overall torque reduction gear ratio of 1.7 .
The outside and inside views of this design can be seen in figures 6.12a and 6.12b respectively.



(a) Final LIM version 2.0 and wheel assembly outside view



(b) Final LIM version 2.0 and wheel assembly inside view

Figure 6.12: LIM version 2.0 and wheel design

It was found that the perspex greatly improved the LIMs as they no longer degraded or loosened around the axles. The perspex also stopped the problem of skew gears owing to warping. This iteration of the build of the LIM itself was deemed satisfactory. Alignment of the gears was aided by putting hardboard ring shaped spacers that fit over the nuts between the gears and the bracket. These spacers are flush against the sides of the gears and the spacers to stop any wiggle in the gears. The LIM version 2.0 design does not allow the wheels to be held in place with spacers in the same manner as the gears. This is owing to the practical issues of needing access to the nuts next to the wheel to tighten them. This is not ideal as it allows for wobbling of the wheel. The exploded view of LIM version 2.0 can be seen in figure 6.13 which is intended to aid in understanding the way in which the LIM is assembled.

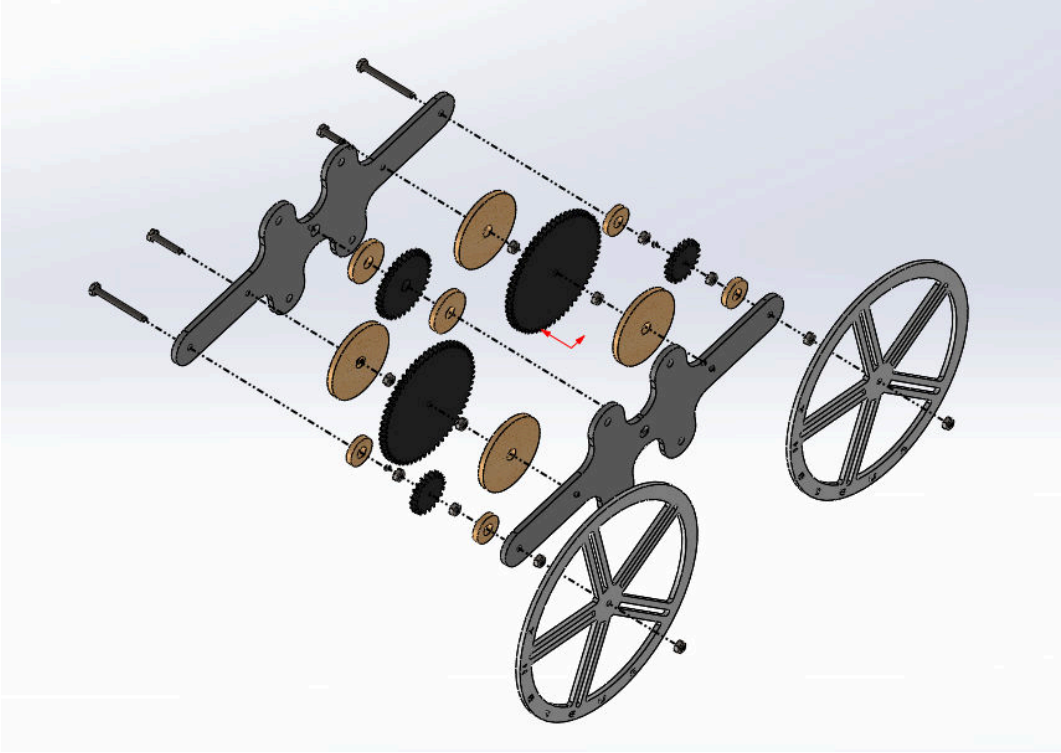


Figure 6.13: Final LIM version 2.0 and wheel design exploded view for assistance in assembly

The wheels shown in figure 6.13 are single 3mm thick wheels however more wheels can simply be glued together and treated as if they are the 3mm thick wheel in he figure.

In order to fix the issue of the wheels wobbling, two attempts were made one can be seen in the following paragraph titled LIM version 3.0. The other attempt involved attaching third bracket to the assembly of LIM2.0 on the outside of the wheel to hold everything more firmly in place. This design is labeled LIM2.1 and can be seen in an exploded view shown in figure 6.14. This solution was effective and stopped the wheel and gears from wobbling. This design was considered a success as it fulfilled the milestone criteria. LIM version 3.0 was still pursued in order to test the effectiveness of climbing obstacles in a different manner.

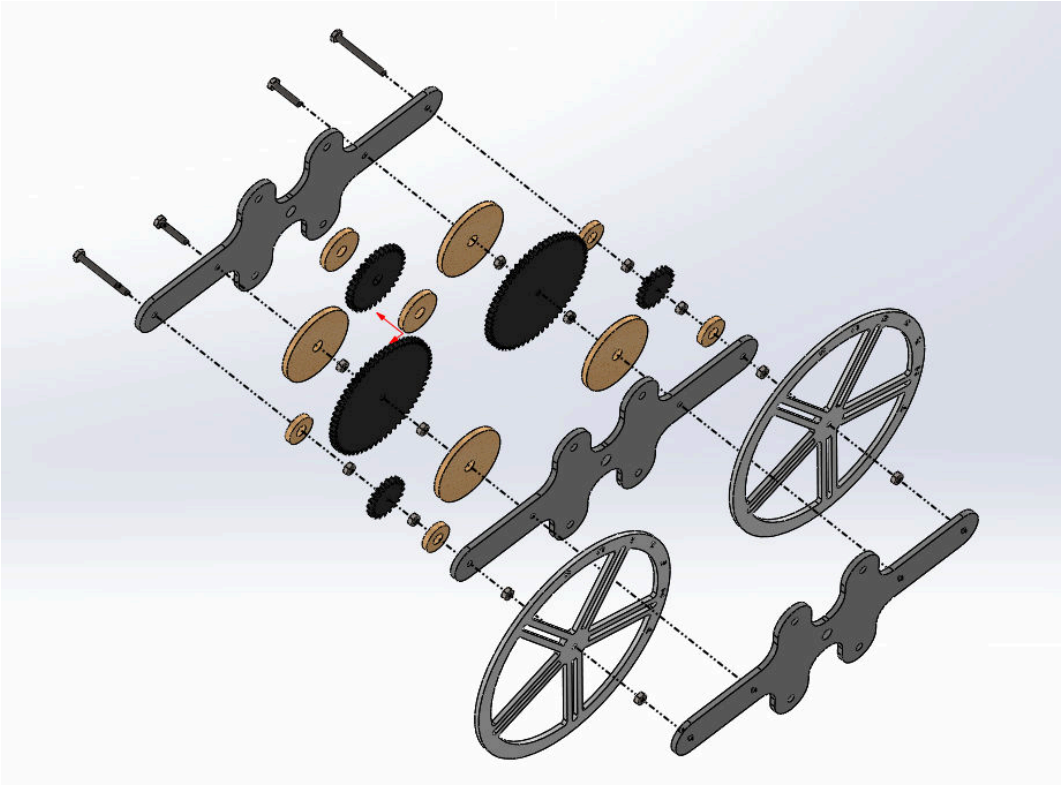


Figure 6.14: Exploded view of LIM version 2.1 to assist in visualisation and assembly

The wheels shown in figure 6.14 are single 3mm thick wheels however more wheels can simply be glued together and treated as if they are the 3mm thick wheel in he figure.

LIM Version 3.0

A solution that has the gears and wheels on the inside of the brackets was pursued to solve the problems of wobbling. This was not possible with the current design making use of an idler gear as the axle of the idler gear obstructs the spokes of the wheel. The idler gear was discarded and the driven and driving gears were made larger. The driven gear was made slightly smaller than the wheel so as not to out-span the wheel. This design resulted in the need to adjust table 6.3 and the design conclusions drawn from it. The adjusted table can be seen in table 6.4 on the following page. The gear labeling shown in the table and any co-ordinate definitions can be seen from the simple kinematics model in figure 6.15.

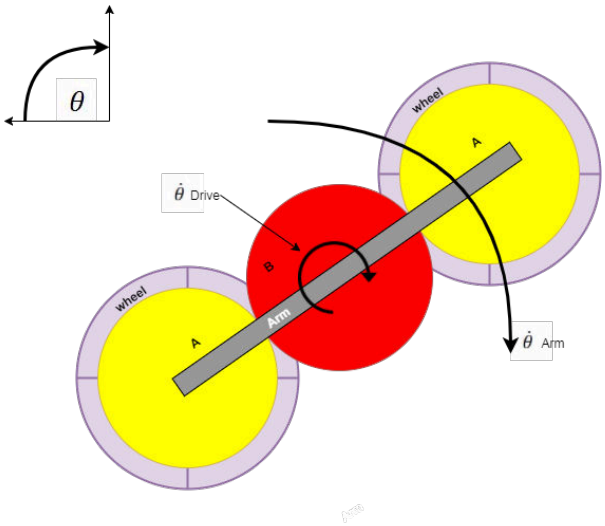


Figure 6.15: Simple schematic for gear labeling and co-ordinate definition for LIM version 3.0

In the calculations to follow generally accepted symbols are used, these symbols and what they represent are:

- θ - Represents the angular position of an object from the negative y-axis measured in degrees as defined in figure 6.9 on the previous page. The clockwise direction is taken as positive by the definition given in figure6.9 on the previous page
- $\dot{\theta}$ - Represents the rate of change of the angular position of an object in degrees per second. The clockwise direction is taken as positive by the definition given for θ

Table 6.4: Motion analysis of LIM version 3.0

	A	B	Carrier Arm
motion relative to frame	$\dot{\theta}_{Wheel}$	$-\frac{N_A}{N_B}\dot{\theta}_{Wheel}$	$\dot{\theta}_{Arm}$
motion relative to carrier arm	$\dot{\theta}_{Arm}$	$\dot{\theta}_{Arm}$	0
Absolute motion	$\dot{\theta}_{wheel} + \dot{\theta}_{arm}$	$\dot{\theta}_{Arm} - \frac{N_A}{N_B}\dot{\theta}_{Wheel}$	$\dot{\theta}_{Arm}$

From this table analysis of the gear linkage is possible. Various conclusions can be drawn about the effect of the gear ratio that allows for analysis of the gear linkage.

- The drive speed is equal to the absolute speed of Gear B as the motor drives gear B directly, Thus:
$$\dot{\theta}_{Drive} = \dot{\theta}_{Arm} - \frac{N_A}{N_B}\dot{\theta}_{Wheel} \tag{6.7}$$
- If the carrier arm is not rotating (Robot is driving forwards) $\dot{\theta}_{Arm} = 0$
$$\dot{\theta}_{Drive} = -\frac{N_A}{N_B}\dot{\theta}_{Wheel} \tag{6.8}$$
- If the wheels are not moving (The wheel is stuck) $\dot{\theta}_{wheel} + \dot{\theta}_{arm} = 0$
$$\dot{\theta}_{Drive} = \dot{\theta}_{arm}(1 + \frac{N_A}{N_B}) \tag{6.9}$$
$$\dot{\theta}_{arm} = \frac{\dot{\theta}_{Drive}}{1 + \frac{N_A}{N_B}} \tag{6.10}$$

Equation 6.10 bears importance on the design and can guide the choice of gear ratio. This equation helps identify the effect that various gear ratios will have on the robot while it is driving. From equation6.10 it can be seen that there is no longer a change in the sign of the fraction on the right hand side of the equation depending on the gear ratio, this indicates that different gear ratios will no longer result in a change in the direction of the rotation of the arm. The conclusions drawn are:

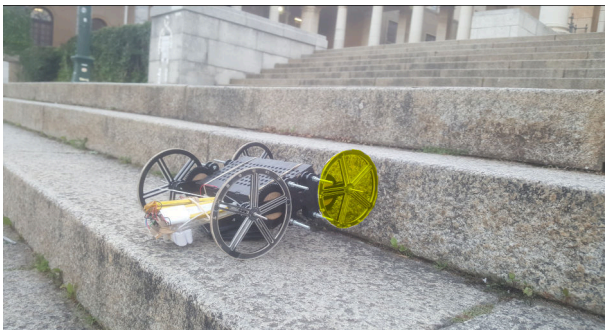
- For all relationships of N_A and N_B:
The arm rotates in the same direction as the drive but opposite to the direction of the wheel. This is the desired motion for climbing objects and steps through rolling the front wheel up the front of the object when the front wheel becomes stuck.
- There is no longer a gear ratio where:
The drive is not capable of rotating the arm. No work can be done by the drive at this gear ratio.

- There is no longer a gear ratio where:
  The arm rotates in the direction of the wheel, this would help the robot overcome small objects by rotating the rear wheel over the front wheel and on to the obstacle.

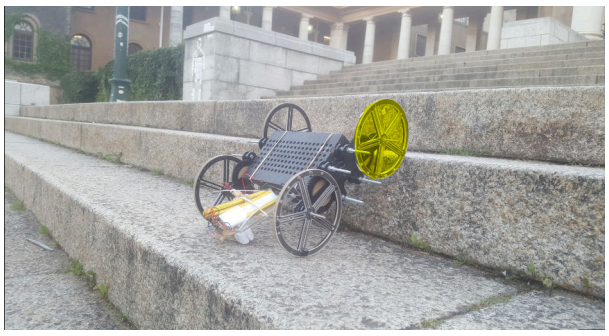
Therefore from this analysis it can be seen that the robot is desired to climb by rotating the arm of the LIM so that the front wheel rolls up the obstacle rather than rotating the rear wheel over the front wheel. This was not the initially intended design but was pursued, as if it gets over the obstacles in question is is a successful design of a robot that can overcome obstacles.

This motion desired from LIM version 3.0 can be seen in figure 6.16 on the following page. Figure 6.10 is made up of six figures that are presented in the order that the motion should occur, starting with figure 6.16a indicating the front wheels coming into contact with the obstacle. Figure 6.16b follows showing the front wheels beginning to roll up the step, this is followed by the front wheels reaching the top of the step in figure 6.16c. Figure 6.16d the front wheels coming over the obstacle and figure 6.16e shows the robot lifting itself up by using the front wheels which are now on top of the obstacle. Figure 6.16f shoes the robot completing its motion by overcoming the obstacle. In these figures the wheels that were at the front, at the start of the motion is highlighted throughout the motion to allow the progression of the motion to be easily followed.

(a) step 1 of the desired motion for driving the front wheel on to the obstacle

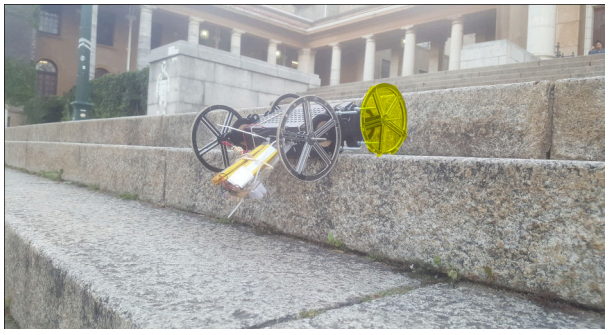


(b) step 2 of the desired motion for driving the front wheel on to the obstacle

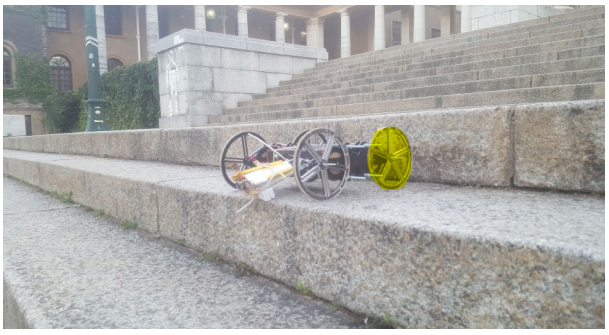


(c) step 3 of the desired motion for driving the front wheel on to the obstacle

(d) step 4 of the desired motion for driving the front wheel on to the obstacle



(e) step 5 of the desired motion for driving the front wheel on to the obstacle



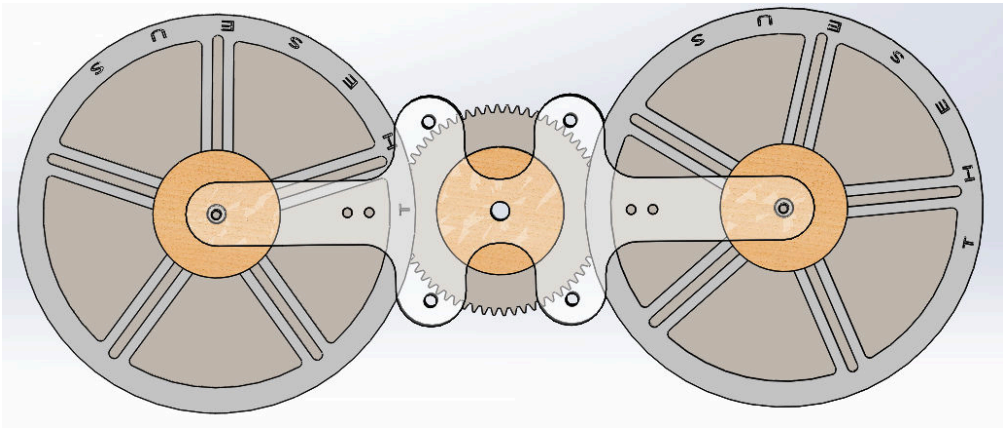
(f) step 6 of the desired motion for driving the front wheel on to the obstacle

Figure 6.16: Demonstration of steps for the desired motion to drive the front wheel on to an obstacle

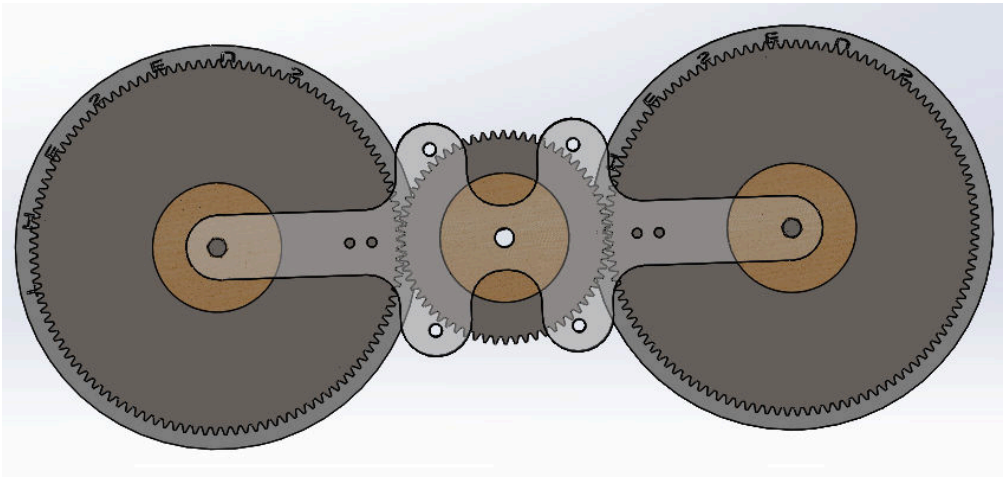
Using the information in section 6.1.4 and table 6.4 the gears were redesigned. The gears designed had the following number of teeth:

- 64 teeth on the center driver gear (paired to the motor)
- 114 teeth on the driven gears (paired to wheel)

Using equation 6.6 and knowing that the distance between the central hole and the wheel hole on the LIM bracket is 89mm, it can be confirmed that
$\frac{114}{2} + \frac{64}{2} = 89$
It was also ensured using a Solidworks model that these gears would not collide with the bolts holding the LIM brackets together. the front and back views of this LIM assembly can be seen in figures 6.17a and 6.17b respectively.



(a) Final LIM version 3.0 and wheel assembly outside view



(b) Final LIM version3.0 and wheel assembly inside view

Figure 6.17: Final LIM version 3.0 and wheel design

An exploded view can also be seen in figure 6.18 to assist in understanding of how the LIM and wheel structure is assembled.

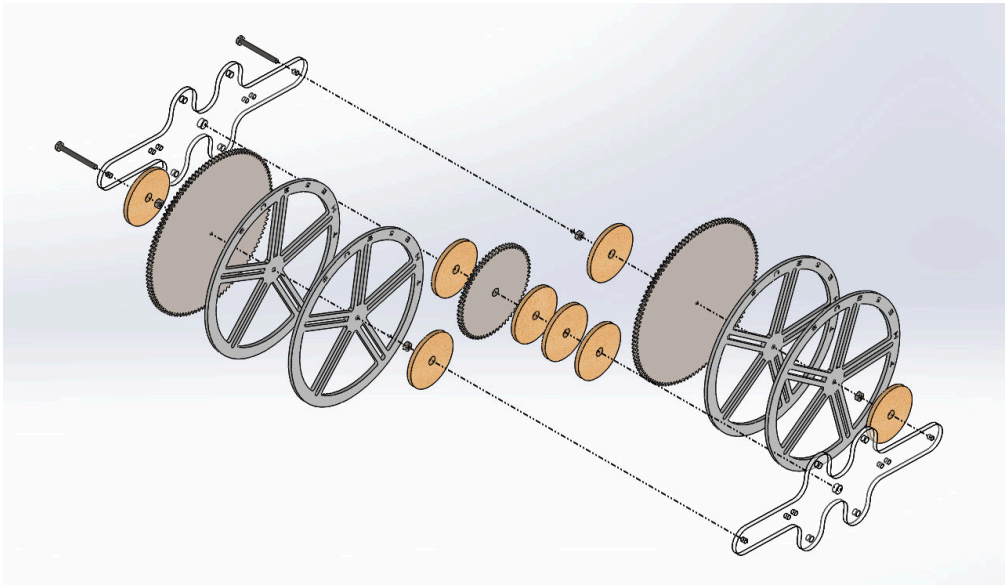


Figure 6.18: Final LIM version 3.0 and wheel design exploded view for assistance in assembly

Solidworks Simulation

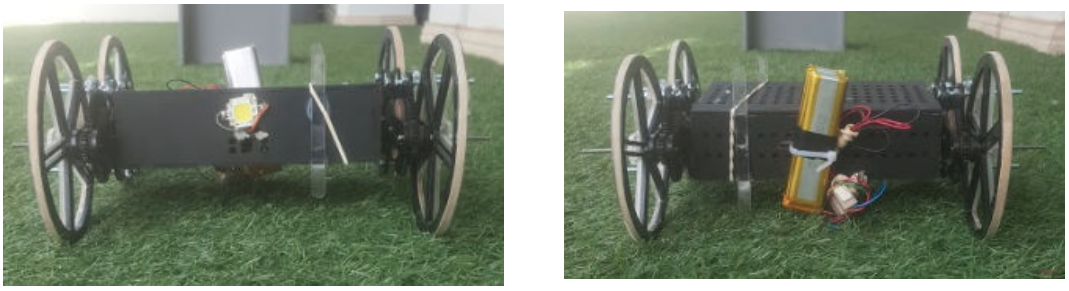
LIM versions 2.0 and 3.0 were designed fully in Solidworks. In order to simulate the operation of the gear mechanisms and confirm that they would act as desired, the LIM bracket was set as fixed and the gears were rotated. It was observed that the wheels rotated as desired. The LIM bracket was set to float and the front wheel was set as fixed and the gears were rotated. It was observed that the whole LIM structure rotated as desired. Through this simulation it was confirmed that the LIMs would function as desired if enough torque was supplied and if the wheels became adequately stuck on obstacles without slipping.

6.1.6 Counter Spin Mechanics

by considering the torques in the system it can be seen that without some form of counter torque arm on the main body, the body will spin rather than the Gears or LIM. This is because the length of the body provides a shorter torque arm than the wheel base making it an easier object to spin.

Stopper Rod Design

A simple mechanical stopper rod design was investigated in order to stop the body from rotating on its axle rather than spinning the gears or LIMs. These rods simply provide a mechanism to stop the body from rotating past a certain point and can be easily attached wherever needed. The rods are not expected to provide a lot of drag on the body however if drag becomes a noticeable problem small wheels can be attached to the bottom of the rods. In order to facilitate this design the battery box can be mounted directly to the main body rather than having the tail. The rods used were pieces left over from the laser cutting over other pieces of the design and therefore no Solidworks design was one however the realisation of this design can be seen in figure 6.19.



(a) Stopper rod attached to front of robot

(b) Stopper rod attached to back of robot

Figure 6.19: Implementation of stopper rod solution to front and back of robot.

Figures 6.19a and 6.19b show the implementation of the stopper rod on the front and back of the main body of the robot.

Tail Design

The problem of spinning can be remedied by placing some weight attached to the body at some distance away from the main body. Calculations of the moment that would need to be applied to the body to prevent spinning can be complicated owing to the need to analyse the gear linkages and the torque required to spin the gears and wheels. The weight and distance required can be investigated through experimentation as a hands on trial and error approach will help give a feeling for the moment that is needed. The main contributors of weight to the body are the motors and the batteries. The motors have been placed inside the main body housing to remove the need for complex transmission systems to get torque from some distance away from the central body to the LIMs. For this reason the batteries were chosen to be placed away from the main body in a separate battery housing on the end of what can be referred to as a "tail". Placing batteries far

away with long wires could cause problems to signals however the distance should not be significant enough to cause any noticeable degradation in the performance of the electronics. An example of this design can be seen in figure 6.20.

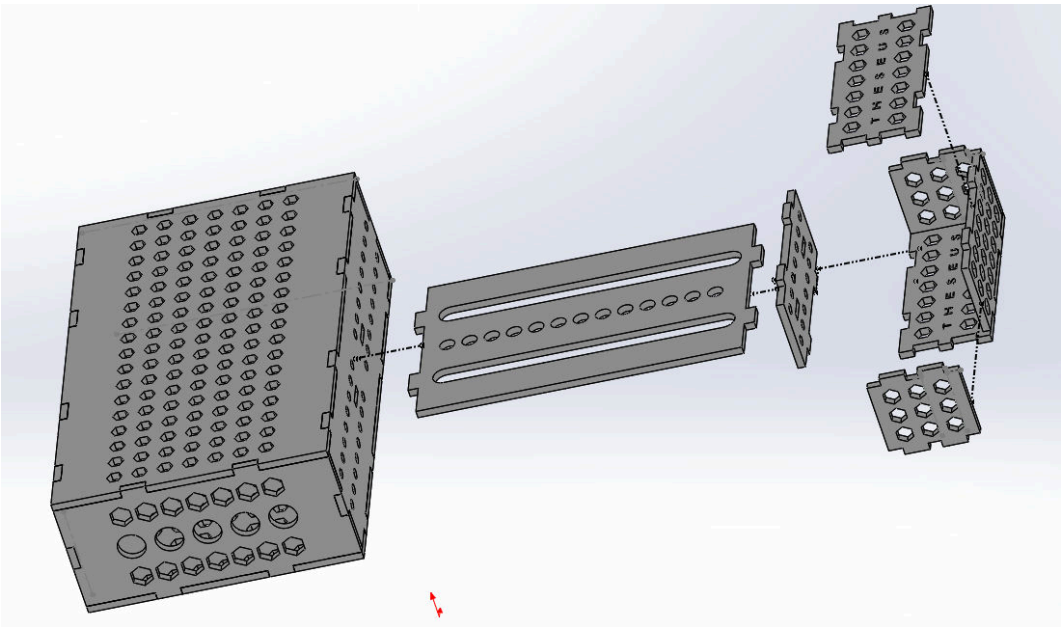


Figure 6.20: Solidworks design for the tail counter spin solution

The design shown in figure 6.20 allows for the robot to still be modular. The tail slides into two grooves in the main body housing and can be glued in place. for extra support I-beams can be placed down the length of the tail for support if needed. If the tail breaks the plugs from the tail can be pushed out of the grooves and a new tail can be inserted and glued in place. The battery housing is designed in the same puzzle like fashion as the main body and can easily be adjusted in Solidworks and laser cut to fit various sized batteries.

6.1.7 Full Mechanical Build

Once all the individual mechanical components were deemed to have met their design requirements they were integrated into the full mechanical system as shown in figure 6.21. Schematics of the various subsystems and the full system can be seen in appendix C. The dotted line in figure 6.21 shows the axis along which the LIMs are mounted on the motors shafts which are secured to the main body.

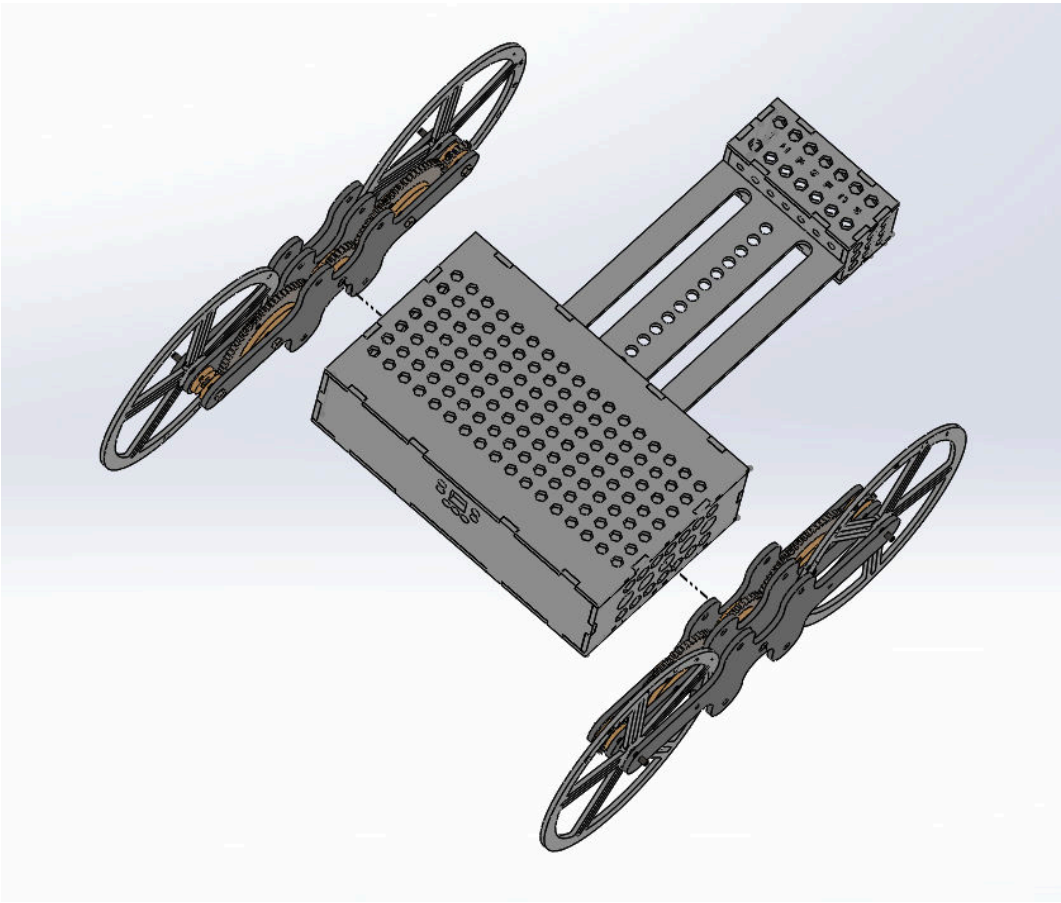


Figure 6.21: Solidworks design for the full mechanical system

6.2 Electronic Deign

6.2.1 H bridge

The robot will require full voltage over each motor selectively in both forward and reverse directions, this can be achieved through the use of a full H-Bridge for each motor. There is a design trade-off between using an existing H-bridge chip like the TB6612FNG which is a double full H-bridge package shown in figure 6.22 (cost 74.95 ZAR) or designing and implementing a new H-bridge using a quad MOSFET package (cost 10.73 ZAR) such as the ZXMHC3F381N8TC package, the layout of which can be seen in figure 6.23. This package would be able to be implemented as a single full H-bridge therefore the total cost on chips for a double H-Bridge would be 21.46 ZAR. The existing H-Bridge chip is small compact and would be simple to implement but designing an H-Bridge from the quad MOSFET chip would be cheaper and if implemented on PCB may not be much bigger than the existing H-Bridge chip. The MOSFET H-bridge could be much more efficient as it uses MOSFETs rather than BJTs. MOSFETs are preferred for high power applications because they are usually more efficient than BJTs. MOSFETS do not have the PN junction voltage drop that BJTs have. This means that in the on state the MOSFET acts as a resistor while the BJT acts like a resistor as well as a voltage drop. The MOSFET H-Bridge was considered a valid design choice as the benefits of cost and efficiency outweigh the time needed for design and implementation. Design time would only need to happen once after which the initial design could then be re-used if multiple robots were to be made, whereas the added cost for each H-Bridge chip would be incurred for each robot built.

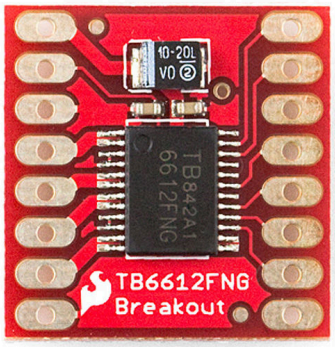


Figure 6.22: TB6612FNG H-Bridge chip.

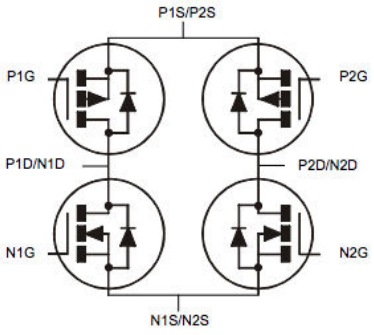


Figure 6.23: Layout of ZXMHC3F381N8TC package.

6.2.1.1 MOSFET Choice

A quad MOSFET SOIC package such as the one shown in figure 6.23 is compact and cost effective. The package contains two P-channel MOSFETS and two N-channel MOSFETS. this will allow for the MOSFETS to be switched as needed for the H-Bridge application. Switching information was found from the data sheet of the package.
P channel MOSFETS in package

- Drain source breakdown voltage: (-30 V)
- Input capacitance: 670 pF @ 1 MHz
- Threshold gate-source voltage: (-1 V) to (-3 V)

N channel MOSFETS in package

- Drain source breakdown voltage: 30 V
- Input capacitance: 430 pF @ 1 MHz
- Threshold gate-source voltage: 1 V to 3 V

The breakdown voltage is acceptable for the applications involved in the robot as the motors are rated 12 V. The other figures will be dealt with in the following paragraphs.

6.2.1.2 Switching MOSFETs

The threshold gate source voltage for the N channel MOSFETs is within the range of the microcontroller outputs however the threshold gate source voltage for the P-channel transistors is not. To ensure that all the MOSFETs are being fully turned on and off a supporting circuit must be included in this design. The supporting circuit will consist of NPN and PNP BJTs that will act to ensure the MOSFETs are receiving enough voltage to successfully switch them.

Switching NPN Transistors
Turning the NPN transistors on and off can be done using the voltage that comes from the microcontroller as the base-emitter threshold voltage is around 0.7V and the emitter is being held at ground throughout operation, therefore a 0v signal from the microcontroller will turn the NPN off and a 3.3 V signal from the microcontroller will be able to turn the NPN on. The configuration to achieve this can be seen in figure 6.24.

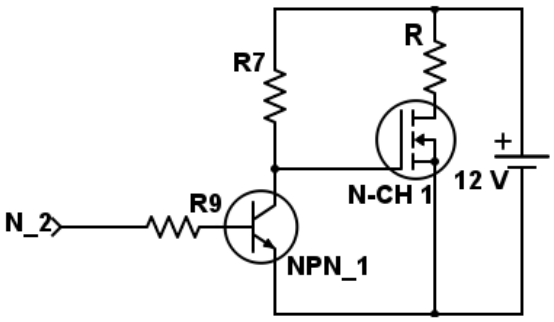


Figure 6.24: NPN BJT connected to fully switch an N-channel MOSFET

Switching PNP Transistors

Turning the PNP transistors on can be done using the voltage that comes from the microcontroller as the base-emitter threshold voltage is around (-0.7 V) and the emitter is being held at Vcc throughout operation, therefore a signal lower than 11.3 V from the microcontroller will turn the PNP on.

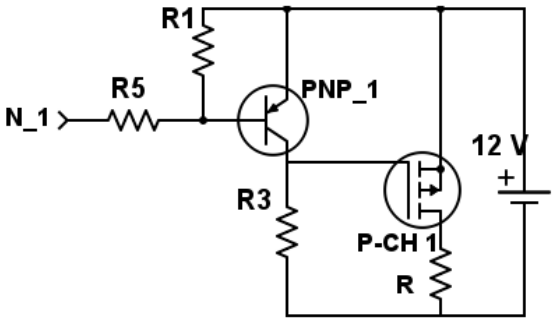


Figure 6.25: PNP BJT connected to fully switch a P-channel MOSFET

Turning the PNP off cannot be achieved by the voltage from the microcontroller as the voltage of the base will need to be above 11.3 V therefore resistor R1 is included as shown in figure 6.25 to create a voltage divider with resistor R5. Figure 6.25 shows how the PNP BJT and the P-channel MOSFET can be connected to ensure correct switching. The values for these resistors are calculated to give the desired response on the base of the PNP transistor of 12 V when the microcontroller outputs 3.3 V (output of microcontroller estimated at 3 V for worst case scenario) and less than 11.3 V on the base of the PNP when the microcontroller outputs 0 V. These are shown in calculations 6.11 and 6.12 When microcontroller output is 3 V:

$$\begin{aligned} V(base) &= V(cc) * \frac{R_5}{(R_1 + R_5)} \\ (11.5 - 3) &= (12 - 3) * \frac{R_5}{(R_1 + R_5)} \\ 8.5 &= 9\frac{R_5}{(R_1 + R_5)} \\ R_1 &= 0.05886 * R_5 \end{aligned} \tag{6.11}$$

Now assume microcontroller current to be 2 mA

$$\begin{aligned} R_5 &= \frac{V(base)}{I_\mu} \\ R_5 &= \frac{11.3}{0.002} \end{aligned} \tag{6.12}$$

R_5 = 5.65 KΩ use(5.6 kΩ) and from 6.11 R_1 = 332 Ω (use 330 Ω) Note: R2=R1 and R6=R5 Using these values in a test circuit on a breadboard the results of which can be seen in table 6.5.

Table 6.5: PNP test with calculated resistor values

V_{cc} [V]	V_{micro} [V]	V_{Base} [V]	$V_{Collector}$ [V]
12	0	11.3	11.9
12	3.3	12	0.1

The values shown in table 6.5 show that the resistor values calculated provide the desired output for the application and can be used. The collector voltage of 100 mV for the 3.3 V base voltage is not ideal however the collector voltage just needs to be below 11.3 V to turn the P channel MSOFET on therefore it is acceptable.
The full circuit schematic of the MOSFETs connected to the supporting circuit is shown in figure 6.26.

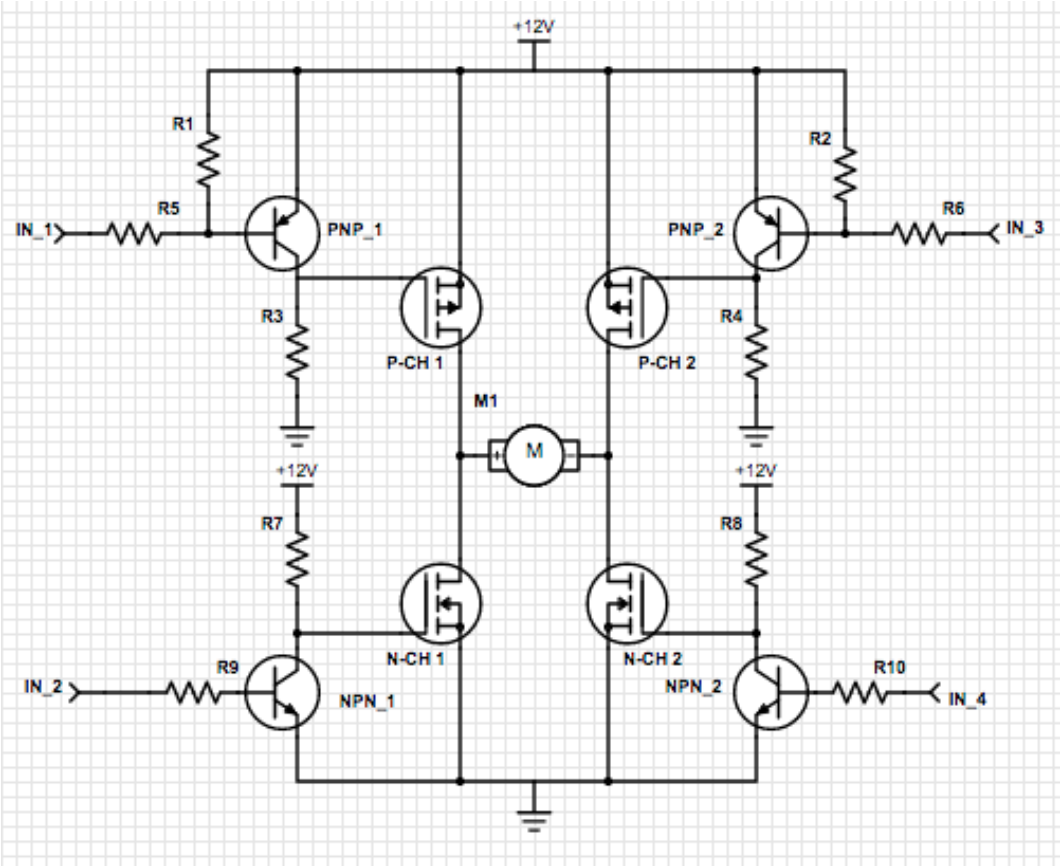


Figure 6.26: Mosfet H-Bridge with supporting circuitry

6.2.1.3 Using the H-bridge

Each H-bridge requires:

- Positive voltage and ground from the battery
- Two lines that switch between 3.3 V and ground to switch the PNP transistors which switch the P channel MOSFTETs
- Two PWM signals (to control speed) to switch the NPN transistors which switch the N channel MOSFETs

The motor can be run by turning on one of the P-channel MOSFETs and supplying a PWM signal to the N-channel MOSFET of the other side of the H-Bridge. This must be done while keeping the other two MOSFETs off to prevent shoot through as shown in table 2.2 in section 2.5.2. The result of this is the P-channel MOSFET staying on and the N-channel MOSFET turning on and off with a desired duty cycle that will result in the motor running at a speed proportional to the duty cycle of the PWM. The direction can be reversed by doing the same on the other P-channel MOSFET and N-channel MOSFET. The PWM signals can be created by using the TIM peripherals on the microcontroller and the digital bits are implemented by setting general purpose input/output pins on the microcontroller to output mode and changing them low or high.
In order to close a P-channel transistor the voltage of its gate must go low relative to its source and in order to close an N-Channel transistor the voltage on its gate must go high relative to its source. Therefore the desired inputs to the BJTs and the MOSFETs can be seen in tables 6.6 and 6.7

Table 6.6: Desired states for H-bridge transistors

	Direction 1		Direction 2	
	Gate Voltage [V]	Open/Closed	Gate Voltage [V]	Open/Closed
P-Chan 1	0	closed	12	Open
P-Chan 2	12	Open	0	Closed
N-Chan 1	0	Open	12	Closed
N-Chan 2	12	Closed	v	Open

Table 6.7: Desired states for supporting BJTs

	Direction 1		Direction 2	
	Base Voltage [V]	Collector Voltage [V]	Base Voltage [V]	Collector Voltage [V]
PNP1	3.3	0	0	12
PNP2	0	12	3.3	0
NPN1	3.3	0	0	12
NPN2	0	12	3.3	0

6.2.1.4 Anti Shoot Through Circuitry

In order to prevent any shoot through the microcontroller is connected to a logic circuit that through NOT and NAND gates will only allow desired patterns of signals to be applied to the H-Bridge so that a shoot through condition cannot be applied to the H-Bridge. This allows for simpler and safer implementation in code where a mistake in code wont cause shoot through. In order to achieve the desired states expressed in table 6.6 and table 6.7 the logic circuitry was designed and simulated using logisim and can be seen in figure 6.27.

The circuit design shown in figure 6.27 is designed to control the logic for one H-bridge

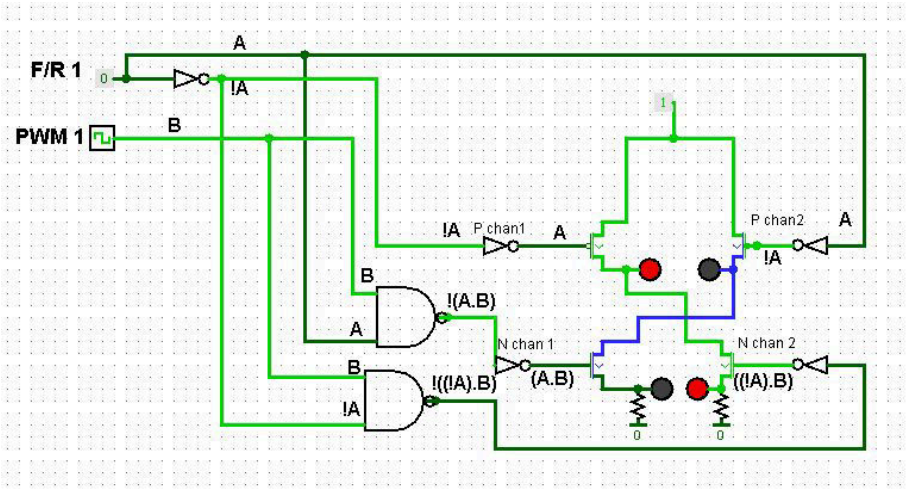


Figure 6.27: Logisim simulation of designed circuit

therefore one of these circuits will be needed for each H-bridge. The not gates prior to the MOSFETs in the circuit simply invert logic just as the stage of BJTs does. In the simulation an on (red) LED indicates an on state on the MOSFET while an off (grey) LED represents an off state on the MOSFET. The Logisim simulation was tested and it showed that no shoot through conditions were applied to the H-bridge for any of the inputs possible. A more in depth look at the simulation can be seen in appendix D. The inputs to this logic circuit, such as the direction bits and PWM signals will come from the microcontroller. The BJTs and MOSFETS from the H-bridge circuitry already discussed

are included in the simulation. Owing to this circuit each H-bridge only needs one PWM signal from the microcontroller as well as only one direction bit instead of two PWM signals and two direction lines. The use of less signals to run the H-bridge allows for less pins to be used on the microcontroller and simpler implementation in code. Once the correct operation of the circuit was confirmed by simulation the circuit was designed to be implemented on veroboard as seen in figure 6.28. The final implementation of the design can be seen in figure 6.29.

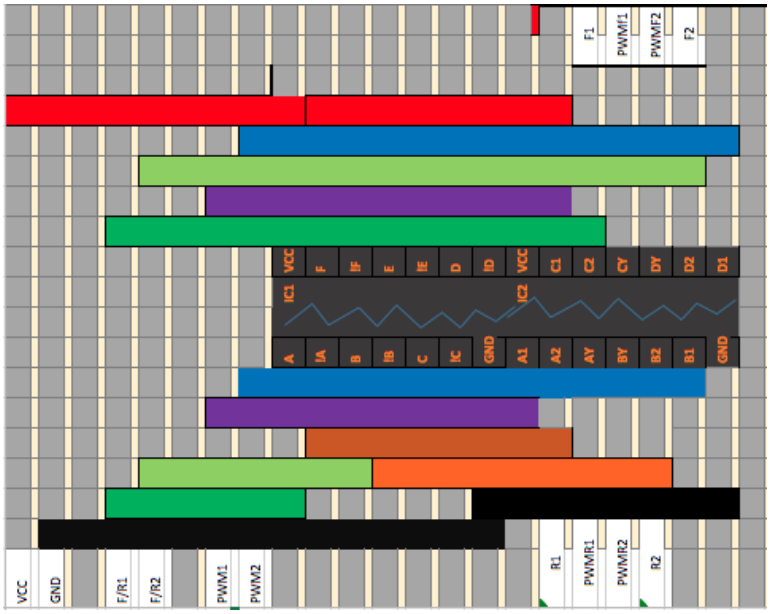


Figure 6.28: Veroboard design for anti shoot through logic circuit.

In the veroboard design in figure 6.28 The white blocks with writing represent molex pins. The black boxes represent ICs (Integrated Circuits) where IC1 is a HC7404, 14 pin, NOT gate chip and IC2 is a HC7400, 14 pin NAND gate chip. The coloured lines represent wires.

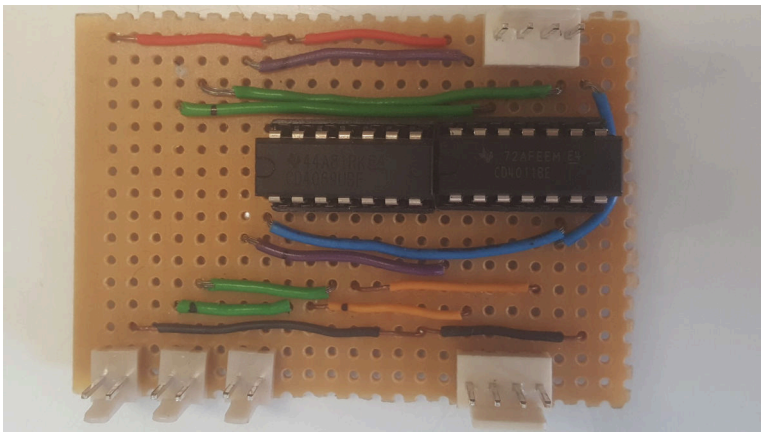


Figure 6.29: Veroboard implementation for anti shoot through logic circuit.

6.2.1.5 Selecting a PWM Frequency

The PWM frequency must be selecting by considering the input capacitance of the N-channel MOSFETs and the Resistor in series with it namely R7 (or R8). The resistor values for R7 and R8 can be calculated as shown in calculation 6.13. It must be stated that the desired collector emitter current of the NPN BJTs is 10 mA

$$R_7 = \frac{V(cc)}{I(ce)} = \frac{12}{0.01} = 1.2k\Omega \tag{6.13}$$

Therefore the RC time constant of the N-channel MOSFETs can be calculated as 516x10^{-9}seconds. This means it will take any transients 4τ=4x(RC) seconds to reach their steady state. Thus the minimum period of the PWM and the maximum frequency can be seen in calculations 6.14 and 6.15

$$T(min) = 4 * (516x10^{-9}) = 206.4x10^{-9} \tag{6.14}$$

$$F(max) = \frac{1}{T(min)} = 484kHz \tag{6.15}$$

Therefore 10 KHz is a safe frequency to choose for the PWM signal to drive the H-bridge.

Owing to time constraints and waiting time for components the MOSFET H-Bridge was not able to be used during testing of the robot. The TB6612FNG was used instead however the design for the MOSFET H-Bridge was simulated as shown in appendix D the simulation confirmed that the outputs were as desired for the given inputs and that the circuit would operate as intended. Based on the simulation an the careful picking of components based on their ratings it would be expected that the designed H-bridge would have fulfilled the required function given more time.

6.2.2 Accelerometer

The MPU-9255 chip as seen in figure 6.30 was used as an accelerometer as it was readily available for use. The MPU-9255 also provides great advantage in that it has an accelerometer as well as a gyroscope, a magnetometer and a temperature sensor which can easily be accessed once the device is communicating with a master device already, this allows for easy expansion and improvement of the robot including being able to provide the operator with information such as heat or the angle that the robot is sitting at. The MPU-9255 is able to communicate with I2C or SPI which has multiple trade offs some of which are shown in table 6.8.

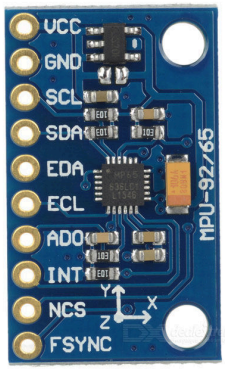


Figure 6.30: MPU-9255 9 degrees of freedom IC.

Table 6.8: I2C Vs. SPI

Communication Protocol	Lines needed	Speed	Ease of use
I2C	2	Slower	Complicated
SPI	3+n*	Faster	Easier

n*= the number of slave devices being attached to the master. For the application of the chip in consideration speed was not considered a relevant factor as a delay of a few microseconds or milliseconds would not be noticeable to the operator. Ease of use is considered more important than the number of lines needed because of the time pressure. SPI is the preferred choice under the circumstances. However with the design philosophy of modularity and easy expansion I2C would be a recommended improvement as it uses fewer lines on the microcontroller allowing more pins to be free for future additions.

The code used to initialise the communication protocol and then read from or write to registers on the accelerometer with the microcontroller were adapted from code given on GitHub by user DoYeouKu [30]. These functions were then used to communicate with the accelerometer. In order to check that the code was reading from the accelerometer the value of the WHOAMI register was read. The WHOAMI register is a register on the slave device that in practice has a reset value not equal to 0x0 and can therefore be used to check if reading from the slave device is being done successfully. Once it is confirmed that the correct value was being read the ability to write was checked by writing an arbitrary value to a particular register on the accelerometer and then reading the value in that register to confirm that it had been written correctly. Once it was confirmed that communications had been set up with the accelerometer, code was written to read the

vertical acceleration from the accelerometer. various values were recorded at different orientations to see what the range of value outputs from the accelerometer looked like. These values can be seen in table 6.9

Table 6.9: Output values for various orientations of accelerometer

Orientation	Output value
Right way up	>10 000
Upside down	<-10 000
Sideways	+-0

Using these values thresholds were set for various conditions such as the accelerometer being upside down the accelerometer being right way up and the accelerometer being on its side. A dead band was also created for when the accelerometer is near being sideways so that the accelerometer does not jitter between outputting upside down and right way up when it is on the verge of being between the two, this was also done to reduce the effect of shaking on the final outcome of calculating its orientation from the accelerometer. The code used to determine the orientation can be seen in figure 6.31 however the full code can be seen in appendix E

```
for(;;){                        //main loop for code
    sel = GPIO_BSRR_BR_15;
    deselect=GPIO_BSRR_BS_15;

    address=0x40;                                       //high acc register
    value0 =  read_from_address(address, sel, deselect);
    address=0x3F;                                       //low acc register
    value1 =  read_from_address(address, sel, deselect);
    value_full = value1 << 8 | value0 & 1<<16;          //creatiing 16 bit number from two registers
    address=0x40;
    value_16 = read_from_address(address, sel, deselect);
    if (value_full > thresholdup) GPIOB->ODR = 0b1;     //threshold for right way up
    else if (value_full < thresholddown) GPIOB->ODR = 0b10; //threshold for upside down
    else GPIOB->ODR = 0;                                 //display noting if in deadband

}
```

Figure 6.31: Code to determine orientation from accelerometer

6.2.3 RC Communication

The RC communication design was guided by the popular use of hobbyist RC devices. RC communication is widely used and has a large amount of online information and guidance for hobbyists owing to the popularity of the interface. Parts of code for specific functions will be shown throughout this section for illustration and explanation purposes only. The final code can be found in full in appendix E

6.2.3.1 The Remote Control

There are a multitude of radio remote controls, any of which would be suitable given that the receiver chosen communicates as needed with it. These remotes can become very expensive however they are not included in the price of the part of the system that is "disposable". For the purposes of testing the robot a SPEKTRUM DX8 remote control was chosen as it did not need to be purchased and was easily accessible. The DX8 allows for a lot of customization of controls allowing the control system to be easily adapted and designed to be simple and intuitive to the operators preference. The DX8 to be used can be seen in figure 6.32.

Figure 6.32: DX8 remote control

6.2.3.2 The Receiver

The receiver chosen is the OrangeRx R617XL as seen in figure 6.33 owing to its low price, availability and use of PPM communication protocol. The receiver outputs PPM as explained in section 2.5.3. This is beneficial as it minimises the need for multiple pins to be used on the microcontroller and also allows the receiver to be small and lightweight.
In order to verify that what the receiver was outputting was of the form expected. The receiver was bound to the remote and then the signal line was connected to an oscilloscope. The normal waveform when the remote is on but not being used can be seen in figure 6.34.

Figure 6.33: OrangeRx R617XL receiver

Figure 6.34: Normal waveform from RC receiver

Using the oscilloscope cursors the following was found:

- Minimum pulse width: 778μs
- Maximum pulse width: 1600μs
- Length of low between pulses: 300μs
- Length of high between pulse trains: 1319μs
- Length of channel 3 pulse in safe mode*: 670μs

*when the safe mode switch is flipped on the controller it drops the length of channel 3 below what is possible using the joysticks.
Using this information sense can be made of the signals with a microcontroller.

Capturing Pulse Lengths

The PPM signal is fed into a pin (PA8) on the STM32F051C6 microcontroller and is separated into the 6 distinct signals. This is achieved through the use of input capture with TIM1. The process works as follows:

The Timer peripheral is set up to read from PA8. The PPM signal is fed into PA8. The timer creates an interrupt and stores the time value at each rising edge on the signal. These time values can then be stored to variables so they can be used to determine the time between rising edges. This is done by subtracting the previous value stored from the current value being read. Once calculated the lengths of the pulses are stored to an array consecutively. When the pulse length read is long enough to indicate that the long, sequence ending pulse has occurred the process starts again from the beginning of the array thereby refreshing the values for each pulse in the array. The values are stored to the array in such a way that channel 1 gets stored to the the 0th element of the array and channel 1 gets stored to the 1st element of the array and so forth. Each of these values in the array can then be used appropriately. The code for calculating pulse length can be seen in figure 6.35. It must be noted that each value stored includes the time of the channel pulse along with the time of the short pause pulse that follows it.

```
void TIM1_CC_IRQHandler(void)
{

    time_old = time_new;
    time_new = TIM1->CCR1;
    if ((time_new-time_old)<2000)           //check if pause pulse or signal pulse
    {
        input_cap [ i ]= (time_new-time_old);   //signal pulse
        i ++;
    }
    else if  ((time_new-time_old)>2000)     //pause pulse
    {
        input_cap [ i ] = 0;
        i = 0;
        input_cap [ i ]= (time_new-time_old);
    }
}
```

Figure 6.35: Code for input capture of pulse width

Using Captured Values

In order to effectively use the captured values the range is mathematically changed to be from -100 to 100 for each channel. This is achieved through the process shown in the first line of code in figure 6.36. Once the range has been altered these values can be used to set the PWM speed based on the magnitude of the value. The duty cycle increases with increasing magnitude of the value. The direction bits can be set by using the sign of the value where negative indicates reverse and positive indicates forward. The code was run and it was observed that the motors did not stop when the controller was in the rest position. This was solved by implementing a dead band around 0 so that the speed is set to zero if the converted values are between -20 and 20 allowing a safe dead band around zero for the motors to stop when the controller is in the rest position. The code to accomplish this can be seen in figure 6.36 and is placed within the interrupt handler shown in figure6.35.

```
right=(input_cap[2]/4)-379;      //signal for right wheel (set from -100 to 100)
if (right>20)                    //sort positive and negative signals
{
    right_speed=right;
    right_dir=0;
}
else if (right<-20)
{
    right_speed=right*(-1);
    right_dir=1;
}
else if (right>-20&&right<20)   //dead band to stop in middle
{
    right_speed=0;
}

left=(input_cap[1]/4)-379;   //signal for left wheel (set from -100 to 100)
if (left>20)                     //sort positive and negative signals
{
    left_speed=left;
    left_dir=0;
}
else if (left<-20)
{
    left_speed=left*(-1);
    left_dir=1;
}
else if (left>-20&&left<20) //dead band to stop in middle
{
    left_speed=0;
}
```

Figure 6.36: Code for using RC receiver values

Stopping Jitter

It was found that at regular intervals the motors would jitter from signals being added to the array of values that did not represent true data from the remote. By adding watch expressions on Attolic with the development board connected to the PC and the RC receiver. It was found that large values were periodically being added to the array these

values were in the range of high millions. In order to prevent this a statement was added to filter out any large expressions from being written to the array, however these values persisted. The code for this initial attempt can be seen in figure6.37. The next attempt to

```
void TIM1_CC_IRQHandler(void)
{

    time_old = time_new;
    time_new = TIM1->CCR1;
    if ((time_new-time_old)<15000)      //filter out the receiver being off
    {
        if ((time_new-time_old)<2000)   //check if pause pulse or signal pulse
        {
```

Figure 6.37: First attempt to stop jitter

filter out these values was based on a simple compare action where values would only be added to the array if three consecutive values read off of each channel were within a specific range (i.e a value within 10 below or above the previous value). Such method may introduce some lag but it was decided that the lag would be preferable to the jitter as the lag was not expected to be noticeable owing to the refresh rate of the signal. This method also failed to solve the problem. The code for this attempt can be seen in figure 6.38. This indicated that the problem may be from overflow of the variables in the array.

```
right_2=right_1;                                //shift old variable
right_1=right_0;                                //shift old variable
right_0=(input_cap[2]/4)-379;                   //signal for right wheel (set from -100 to 100)
if (right_0>(right_1-10)||right_0<(right_1+10)) //check within range
{
    if(right_1>(right_2-10)||right_1<(right_2+10))  //check within range
    {
        right=right_0;
        if (right>20)                             //sort positive and negative signals
        {
```

Figure 6.38: Second attempt to stop jitter

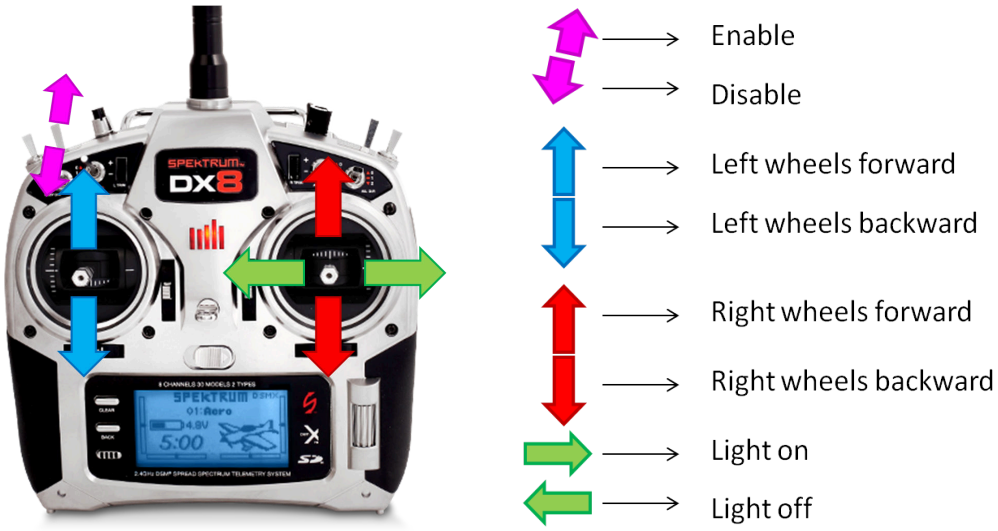
The final solution was to accept that these values could be added to the array but to not allow any value larger than 100 to be accepted into the function which sets the PWM duty cycles. This solution was extremely effective it stopped the jitter and did not cause any lag in the controls. The code can be seen in figure 6.39

```
void set_PWM( int PB_10, int PB_11 )
{
    if (PB_10<100 && PB_11<100)             //filter values larger than 100
    {
        TIM2->CCR3 = (PB_10) * 80;
        TIM2->CCR4 = (PB_11) * 80;          // enable the OC channels
        TIM2->CCER |= TIM_CCER_CC3E;
        TIM2->CCER |= TIM_CCER_CC4E;
        TIM2->CR1 |= TIM_CR1_CEN;           // counter enable
    }
}
```

Figure 6.39: Final attempt to stop jitter

Controller Layout

A system to easily control the robot in an intuitive way which still allowed high maneuverability needed to be developed. The following system was decided upon as it was simple, intuitive and easily implemented with the way data was being received from the remote. The system chosen can be seen in figure 6.40



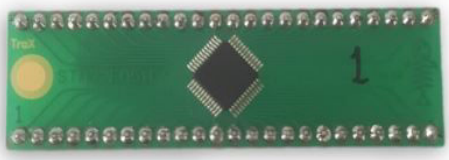
(a) Controller layout (b) Controller key

Figure 6.40: controller format

These controls were found to be highly intuitive and simple, however the functions that specific joystick movements perform can be easily changed in the code. Owing to the use of PPM and storing each pulse length to an item in an array, changes to the controls can be made by simply indexing different items of the array in the code. This allows for easy adaption to the operators needs or preferences.

6.2.4 Microcontroller

The microcontroller is necessary for the implementation of various other electronic components such as the H-Bridge, Radio control and the accelerometer. The microcontroller allows interfacing between these various modules of the design and as such the microcontroller plays an integral part of combining all these modules into a working system. The microcontroller being used as stated in section 2.5.4 is the STM32F051C6 as seen in figure 6.41a



(a) STM32F051C6 microcontroller chip on break-out PCB

(b) UCT development board

Figure 6.41: STM32 microcontroller and UCT development board

In order to use the microcontroller with another module of the robot the module must be connected to the correct pin of the microcontroller and the relevant code must be run on the microcontroller.

6.2.4.1 Running Code

The code can be loaded onto the microcontroller using the UCT development board as seen in figure 6.41b and a PC running Atollic. The code is written in programming

language C and the full code can be found in appendix E. The C code is compiled by Atollic and loaded to the microcontroller by debugging. The microcontroller can then be removed from the UCT development board and placed onto a smaller veroboard designed to hold the microcontroller and connect it to other devices and components. This veroboard is preferable as the UCT development board is large and heavy.

Initialisation

the microcontroller needs to be initialised appropriately for the microcontroller to communicate with external peripheral devices . Initialising the microcontroller involves a number of steps including initialising the pins on the microcontroller to do the specific functions that they are intended to do.

Code Set Up

The bulk of the code being used can be placed within the infinite while loop in the main section of the code. This will include reading from the accelerometer and using the readings from the accelerometer as well as controlling the PWM signals to the motors. The remaining code used will be to interface with the remote control this will be done using input capture which will create an interrupt each time a rising edge is triggered as explained in section 6.2.3.2 . Each time an interrupt is triggered the interrupt handler will be called and run. The interrupt handler should not contain any lengthly code however it can be used to sort and store the information that is being received from the RC controller which is done as explained in section 6.2.3.2. The full code implemented can be seen in appendix E and a flow diagram describing the order that the code occurs in can be seen in figure 6.42 on the following page.

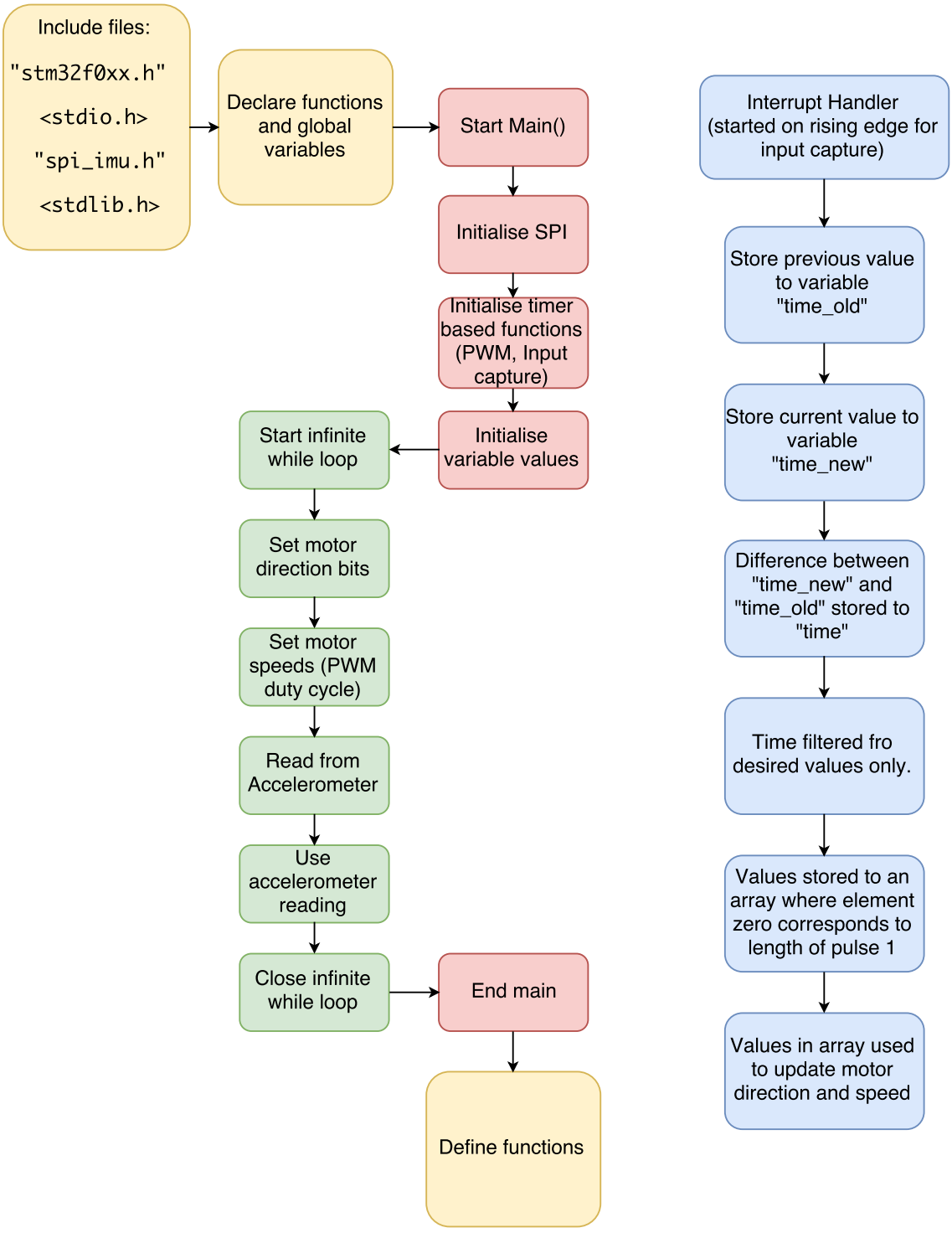


Figure 6.42: Flow diagram for code used

6.2.4.2 Pin Connections

The pin connections to a microcontroller are shown using a hardware systems block diagram. These diagrams are helpful in indicating what modules are being attached to the microcontroller, which pins they are being connected to and how the module and the microcontroller are interfacing i.e logic, I2C, SPI, analogue signals or any other form of information exchange. The hardware systems block diagram for the microcontroller and its connections to various other modules can be seen in figure 6.43. In the figure outputs from the microcontroller are shown as arrows out of the microcontroller and inputs to the microcontroller are shown as arrows into the microcontroller.

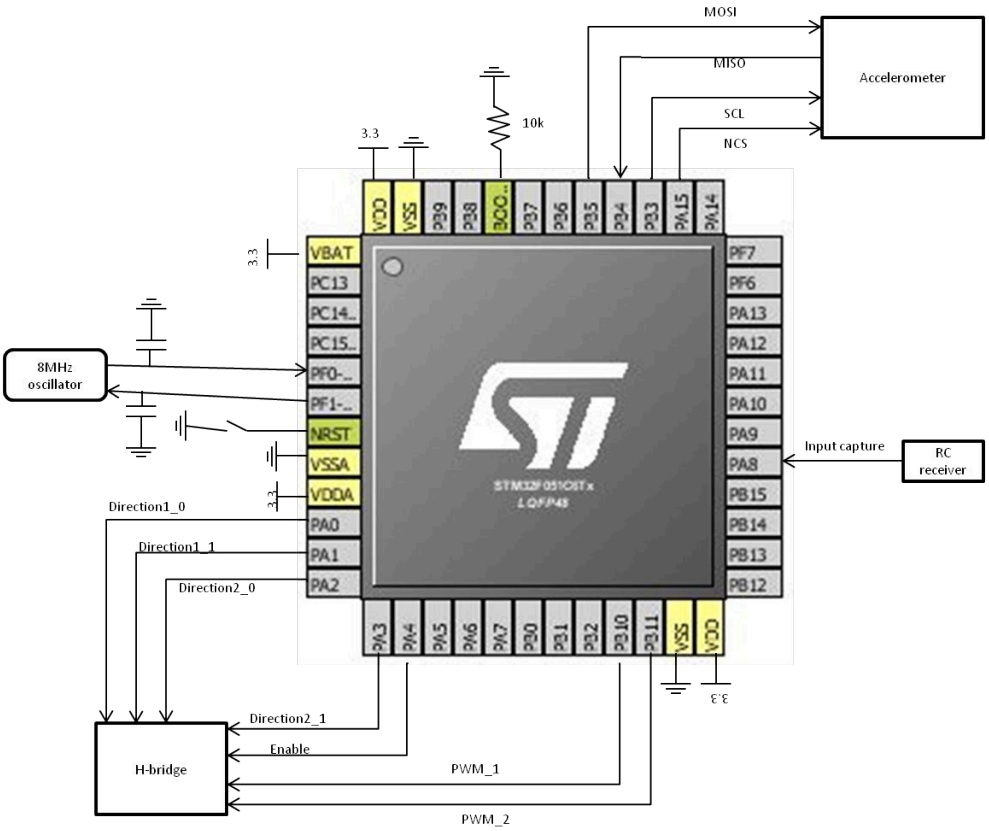


Figure 6.43: Hardware systems block diagram

Figure 6.43 shows all the connections that must be set up with the microcontroller. This diagram can be used when creating a veroboard breakout board for the microcontroller so that the UCT development board does not have to be used. This diagram can also be used to assist with the coding as explained in section 6.2.4.1.

6.2.5 Camera Feed

In order to establish a video feed for the systems requirements an Eachine TX01 camera and transmitter package was selected to be placed inside the robot. A viewing hole was placed in the front of the main body for this purpose. A RC305 receiver was used to pick up the signal from the camera. During implementation the camera had a Right Hand Circular Polarized (RHCP) antenna and no right hand circular polarized receiver antenna was available. A Left Hand Circular Polarized (LHCP) antenna would not be able to receive the transmission at all and a linear polarized antenna would be able to receive the transmission with significant losses. With no other options available the linear polarized receiver antenna was used. Both the camera and the receiver where able to operate off of 5705 MHz, therefore both the transmitter and receiver were set to this frequency and the video transmission system was set up. The receiver has mechanical switches to choose the channel and the camera, transmitter package stores the channel being used on power down therefore once the connection was established the camera and receiver could simply be turned on and off to create and break transmission when necessary. The receiver can be connected to video viewing devices such as screens or video goggles, the FatShark predator goggles were used for testing. The camera and transmitter can be seen in figure 6.44a and the receiver with a linear polarized antenna can be seen in figure 6.44b.

(a) Eachine TX01 camera, transmitter and RHCP antenna

(b) RC305 receiver

Figure 6.44: Video feed receiver and transmitter equipment

The fat shark predator video goggles can be seen in figure 6.45.

(a) FatShark predator goggles outside view

(b) FatShark predator goggles inside view

Figure 6.45: FatShark predator video goggles

7 Test Procedure and Results

A reliable and repeatable test procedure must be used to provide meaningful results from this project. This chapter deals with various suggested test cases in order to ascertain meaningful information about the build of Theseus done in this project. After each test case the results of the test are presented. These results will be analysed and discussed in chapter 8. This information should serve to guide future iterations and improvements on the design by highlighting conditions that effect the success or failure of the design and build.

The testing phase was conducted in an iterative manner. In which the robot is tested and the performance is measured and any changes that can be easily or timeously made in order to improve the robot are made before the final test is completed and the results are recorded. In order to get meaningful results the tests conducted do not have the aim of simply highlighting what the robot is capable of achieving but equally or arguably more importantly they aim to show what the robot cannot do and where its limits lie. By presenting data that shows that the robot could perform a task over and over again it shows robustness and repeatability by not where improvements can be made.

7.1 Effect of Tire Tread on Acceleration and Maneuverability

7.1.1 Test Procedure for Test 7.1

Aim: To determine whether rubber coating the wheels makes a significant difference to both acceleration and maneuverability.
Justification: The robot will need to overcome various obstacles and paths, in order to achieve this the robot must perform as well as possible. The effect that rubber coating the wheels can have on the robots performance must be investigated in order to improve performance as much as possible.
The wheels are laser cut from perspex the possibility of a rubber layer coating them can be explored. Different materials could offer an improvement in speed and turning owing to the different coefficients of friction as explained in section 6.1.2.

7.1.2 Test Procedure for Test 7.1:Acceleration

Aim: To determine whether rubber coating the wheels makes a significant difference to acceleration.

Justification: Lack of traction and spinning of wheels will make the robot difficult to control and it will waste time and battery.
Procedure: Investigating the difference in acceleration in a reliable an repeatable manner is complicated using the on board accelerometer. For this reason this test will be simple and crude, but reliable and repeatable.
The difference that the wheel material makes on acceleration will be determined by:

- setting the robot to full speed from rest
- setting the robot to full speed forwards from 40% speed backwards.
- setting the robot to full speed forwards from moving backwards at full speed.
- setting the robot to full speed backwards from 40% speed forwards.
- setting the robot to full speed backwards from moving forwards at full speed.

Each of these cases will be performed with no rubber on the wheels and then with rubber on the wheels. A simple log of whether slipping occurred or not should be kept.
No degree of slipping will be taken into account (unless the difference is extreme) as that could easily lead to misleading data, as the degree of slipping as observed by a person is objective. Given more time to test, straight line timed tests can be completed to see how much spinning of the wheels occurs when trying to get the robot to move a specified distance in the shortest time possible.

7.1.3 Results for Test 7.1: Acceleration

The results of the test to determine the effect of rubber coated wheels on acceleration can be seen in table 7.1

Table 7.1: Results log for wheel coating effect on slipping

Test	Plain perspex (slip/no slip)	Rubber (slip/no slip)
full speed from rest	slip	no slip
full speed slow reverse	slip	no slip
full speed from full reverse	slip	no slip
full reverse from rest	slip	no slip
full reverse from slow forward	slip	no slip
full reverse from full speed forward	slip	no slip

Turning

7.1.4 Test Procedure for Test 7.1: Turning

Aim: To determine whether rubber coating the wheels makes a significant difference to maneuverability.
Justification: The robot will be required to navigate various different scenarios which may include maneuvering certain paths. It is imperative to know that the robot can be maneuvered and whether rubber coatings will affect this. It is possible that the rubber improves turning ability owing to the grip and the increased predictability and control that it provides. It is also possible that grip will decrease turning ability as it will not allow wheels to slip making turning one side of wheels difficult while the other side stays stationary.
Procedure: The difference that the wheel material makes on turning will be determined by creating paths of desired shapes on the floor using tape. The robot will then be guided along the paths with no rubber on the wheels and then with rubber on the wheels. One operator will perform the test twice without the rubber and twice with the rubber then another operator will do the same. changing the operator and should reduce the effect of the operators capability on the data. A log should be kept where the number of times that the robot went outside the path on each run will be recorded as well as the time taken to complete the path.

7.1.5 Results for Test 7.1: Turning

For the purposes of the test two short differently shaped courses were set up and can be seen in figure 7.1.

(a) Track 1 for testing maneuverability (b) Track 1 for testing maneuverability

Figure 7.1: track for testing maneuverability

The results of the test to determine the effect of rubber coated wheels on maneuverability can be seen in table 7.2 and 7.3

Table 7.2: Results log for wheel coating effect on maneuverability (log for no rubber on wheels)

Operator	Attempt no.	Track 1 (figure 7.1a) (mistake count)[Time]	(Track 2 figure 7.1b) (mistake count)[Time]
Operator 1	1	(1) [19:86]	(2) [14:29]
Operator 1	2	(2) [9:79]	(1) [17:69]
Operator 2	1	(2) [15:65]	(1) [10:10]
Operator 2	2	(1) [9.78]	(1) [7:50]

* all times in table 7.2 are expressed in [seconds : milliseconds]

Table 7.3: Results log for wheel coating effect on maneuverability (log for rubber on wheels)

Operator	Attempt no.	Track 1 (figure 7.1a) (mistake count)[Time]	Track 2 (figure 7.1b) (mistake count)[Time]
Operator 1	1	(1) [8:02]	(2) [6:78]
Operator 1	2	(1) [7:75]	(0) [5:53]
Operator 2	1	(3) [13:00]	(2) [4:03]
Operator 2	2	(1) [7:13]	(0) [3:57]

* all times in table 7.3 are expressed in [seconds : milliseconds]

The tests were completed in the following order:

- track 1, no rubber, attempt 1
- track 1, rubber, attempt 1
- track 1, no rubber, attempt 2
- track 1, rubber, attempt 2
- track 2, no rubber, attempt 1
- track 2, rubber, attempt 1
- track 2, no rubber, attempt 2
- track 2, rubber, attempt 2

7.2 Comparison of Counter Spin Mechanisms

7.2.1 Test Procedure for Test 7.2

Aim: To determine the counter spin mechanism with the best ability to stop the main body from spinning rather than driving the wheels or LIMs.
Justification: The robot needs to be able to transfer the rotational movement from the motor to turning the wheels and LIMs. The body is only attached to the wheels and LIMS with the motor shaft and the body is likely to rotate on the shaft rather than rotating the gears and wheels. To stop this two possible solutions have been isolated in section 6.1.6. Tests must be conducted to assess the effectiveness of the proposed solutions. The results of these tests should indicate whether either or both of the solutions are viable options and indicate the better solution.
Procedure: The effectiveness of the counter spin mechanisms can be tested by attaching the mechanism in question to the body of the robot as described in section 6.1.6. The robot can then be placed on the ground in a obstacle free environment. The following cases were thought to be helpful in assessing the mechanisms ability to prevent rotation.

- Start robot moving forward gradually from rest
- Start robot moving backward gradually from rest
- Start robot moving forward rapidly from rest
- Start robot moving backward rapidly from rest
- gradually change direction of robot from forward to backward
- Rapidly change direction of robot from forward to backward

First each of these cases is tested without a counter spin mechanism in order to determine if the body spinning is an issue that needs to be addressed. Each of these cases can be tested with the counter spin mechanisms to see the changes that occur. A test with added weight to the tail can also be conducted if the tail mechanism with just batteries fails. The results of the test should be recorded in a log.
The words used in the log can be defined as follows:

- Spin is defined as the body of the robot rotating around the axis of the motors shafts rather than the wheels and gears rotating to cause the robot to drive.
- Flip is defined as the robot's main body spinning 180^{o} and then continuing to rotate the gears and wheels to drive.
- Gradual implies movement of the controller joystick from around 40% speed in one direction to about 40 % speed in the opposite direction over a duration of around 2 seconds. Gradually from rest it is defined as moving the joystick from the zero speed position to the 80% speed position over 2 seconds.

- Rapid is defined as movement of the controller joystick from full speed in one direction to full speed in the other direction as quickly as possible. Rapidly from rest is defined as moving controller joysticks from the zero speed position to full speed position as quickly as possible.

7.2.2 Results for Test 7.2

The results to check whether the main body spinning in the absence of a counter spin mechanism can be seen in table 7.4. The results of the test to determine the effectiveness of the different counter spin mechanism designs can be seen in table 7.5.

Table 7.4: Results log for effectiveness of counter spin solutions

Test	No mechanism (Spin/No spin)
Gradually forward from rest	Spin
Gradually backward from rest	Spin
Rapidly forward from rest	Spin
Rapidly backward from rest	Spin
Gradually forward from backward	Spin
Gradually backward from forward	Spin
Rapidly backward from forward	Spin
Rapidly backward from forward	Spin

Table 7.5: Results log for effectiveness of counter spin solutions

Test	Tail (without weight) (Spin/No spin)	Tail (with weight) (Spin/No spin)	Spikes
Gradually forward from rest	No spin	No spin	No spin
Gradually backward from rest	Flip	No spin	No spin
Rapidly forward from rest	No spin	No spin	Spin
Rapidly backward from rest	Flip	No spin	Spin
Gradually forward from backward	No spin	No spin	Spin
Gradually backward from forward	Flip	No spin	Spin
Rapidly backward from forward	Flip	No spin	Spin
Rapidly backward from forward	Flip	Flip	Spin

7.3 Effect of Obstacle Surface on Climbing

7.3.1 Test Procedure for Test 7.3

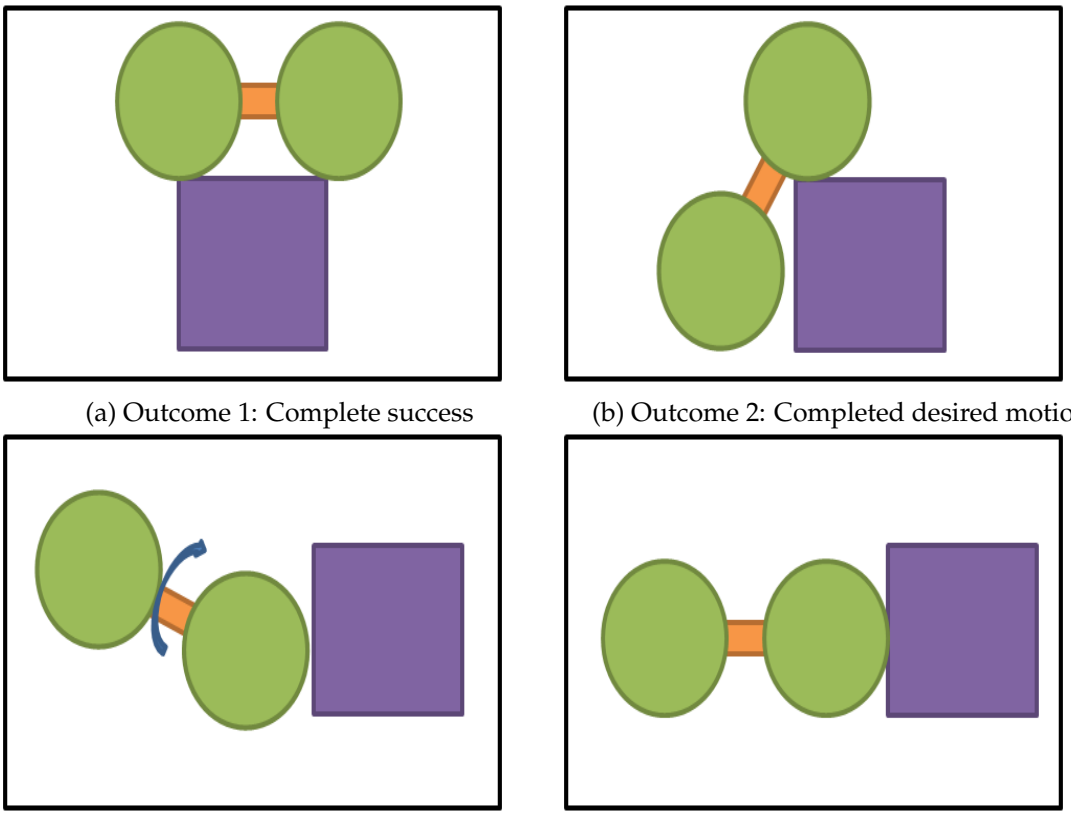
Aim: To determine the robots ability to overcome obstacles made of various materials.
Justification: The robot will need to be able to overcome various obstacles in practice. The material that obstacles will be made of cannot be predicted owing to the nature of USAR, therefore the abilities and limitations of the robot must be investigated.
Procedure: As in section 7.1 this test is concerned with determining the effectiveness of various materials however in this test the effect of the material of the surface being climbed is to be investigated. For this test the obstacle must be covered in various materials the materials to be used will be:

- Wood
- Rubber
- Cardboard
- Sandpaper
- Stone/Concrete
- Perspex

A test must be conducted for each of the above obstacle materials. A log should be kept of each of the attempts and must state whether the robot failed completely, began to climb the obstacle, managed to mount the obstacle or if the robot managed to completely overcome the object. The obstacles in practice will not be able to be picked by the operator for this reason concrete steps or wood steps will be used for all the subsequent test procedures as these will be the most common obstacles in practical situations.
The possible outcomes of the tests can be ranked from 1 to 4 where 1 is the best outcome and 4 is the worst. The outcomes are as follows:

1. Complete success: The robot pulled itself on to the obstacle after completing the desired motion. Seen in figure 7.2a on the following page.
2. Completed desired motion: The robot completed the desired LIM motion but could not pull itself on to the obstacle. Seen in figure 7.2b on the following page.
3. Began the desired motion: The robot began the desired LIM motion but slipping prevented the motion from completing. Seen in figure 7.2c on the following page.
4. Complete failure: The robot did not begin the desired motion owing to the wheels slipping. Seen in figure 7.2d on the following page.

These outcomes can be seen in figure 7.2 on the following page. Figure 7.2 is provided for clarification of what stage in the motion would define which outcome.



(a) Outcome 1: Complete success

(b) Outcome 2: Completed desired motion

(c) Outcome 3: Began the desired motion

(d) Outcome 4: Complete failure

Figure 7.2: Definition of possible outcomes for test 7.3

7.3.2 Results for Test 7.3

The results of the test to determine the effect of an obstacles surface on the robots ability to overcome it can be seen in table 7.6

Table 7.6: Results log for effect of obstacle material and wheel coating

Obstacle Material	Attempt 1	Attempt 2	Attempt 3
Wood	2	2	2
Rubber	2	1	2
Cardboard	2	2	1
Sandpaper	2	2	2
Stone/concrete	4	2	2
Perspex	2	2	4

7.4 Effect of Climbing Motion on Success

7.4.1 Test Procedure for Test 7.4

Aim: To determine the validity of the chosen motion
Justification: The climbing motion chosen for this project is flipping the rear wheels of the robot over the front wheels and on to an obstacle. This is the result of a carefully calculated gear mechanism with some fundamental requirements such as an idler gear and a torque reducing gear ratio. The mechanism to climb in this manner such as LIM2.1 discussed in section 6.1.5.3 is more complicated to design and harder to achieve than the alternate climbing motions such as in LIM3.0 discussed in section 6.1.5.3. For this reason the use of the chosen mechanism must be compared to the use of alternate climbing mechanisms.
Procedure: This test will compare the effectiveness of the back wheel flipping over onto the obstacle motion and the motion of rolling the front wheel up the obstacle. This test can be performed by attaching the LIMs to the robots body and attempting to overcome an obstacle the results should be recorded in a log. If the robot fails to overcome the obstacle a short description of why can be included in the log.
The possible outcomes of the tests can be ranked from 1 to 4 where 1 is the best outcome and 4 is the worst. The outcomes are as follows:

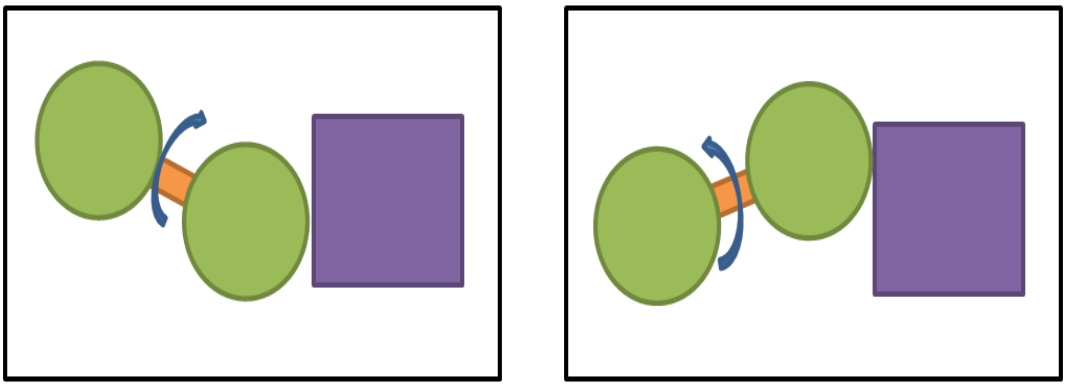
1. Complete success: The robot pulled itself on to the obstacle after completing the desired motion.

2. Completed desired motion: The robot completed the desired LIM motion but could not pull itself on to the obstacle.

3. Began the desired motion: The robot began the desired LIM motion but the motion did not reach the stage where it had mounted the object.

4. Complete failure: The robot did not begin the desired motion owing to the wheels slipping.

Although the motions are different the outcomes are similar to those in test 7.3 and figures for what would constitute each outcome can be seen in the figures shown for test 7.3. Outcomes 1,2 and 4 are the same as those in test 7.3. The difference in the outcomes from those in test 7.3 are because there are now two different definitions for outcome number 3, as the two LIM designs move differently. The two options for outcome 3 can be seen in figure 7.3



(a) Outcome 3 for LIM version 2.1 (b) Outcome 3 for LIM version 3.0

Figure 7.3: Definition of different outcomes for beginning the desired motion

7.4.2 Results for Test 7.4

The results for the the test to determine the effectiveness of the two different climbing motions discussed can be seen in table 7.7

Table 7.7: Results log for effectiveness of climbing motion

LIM design	Attempt 1	Description	Attempt 2	Description
Version 2.1	2	see ** below	2	see **below
Version3.0	3	see *** below	3	see *** below

** : The rear wheel rotated and mounted the obstacle as desired, the wheel was not able to pull the rest of the body up.
***: The front wheel rolled up the face of the obstacle as desired but before mounting the obstacle the LIM continued rotating causing it to then rotate in a complete circle leaving the robot at the starting position having merely swapped the front and rear wheels.

7.5 Duration of Battery Life

7.5.1 Test Procedure for Test 7.5

Aim: To determine the average battery life of the robot in different states of operation.
Justification: The battery life of the robot is an important consideration. It must be determined if the battery meets the specifications set out in chapter 3.
Procedure: For this test the battery life of the robot (in motion and when idle) will need to be calculated. To calculate the expected battery life the robot can be connected to a DC supply which shows current being drawn. A calculation using this current and the mAh rating of the battery will give the expected operation time. the calculation can be seen in equation 7.1.

$$Hours = (mAh\ rating)(Current\ in\ mA) \tag{7.1}$$

The battery life could also be checked by connecting the robot to the battery pack through an ammeter and then following the same process. The use of an ammeter is suitable for checking the current drawn while the system is idle or to check the current drawn by subsystems such as the camera and the light.
To measure the current flowing during normal running operation using an ammeter is not practically possible. In order to check the current being drawn in this case, a battery charger that indicates how much the battery has been recharged is used (a TURNIGY ACCUCELL6 was used). In order to find out how much charge was used the battery is charged to capacity and then the system is run as it would be in practical circumstances. This test was completed over a time span of 2 minutes and a time span of 20 minutes to check whether the results were consistent within reason. This test was run without the light and camera attached as current drawn by the main system, the camera and the light can be stated separately and analysed accordingly.
The charger also displays how long it took to restore the charge, this can be used to calculate how long it would take to fully recharge a battery.

7.5.2 Results for Test 7.5

The results for the tests to determine the current draw of the system while moving can be seen in table 7.8

Table 7.8: Log for current draw during normal operation test

Time used [minutes]	Charge used [mAh]	Average current draw [mA]
2	40	1 200
20	370	1 100

It was decided that the result of the 20 minute test would be more likely to be a true representation of average current draw owing to the greater length of time over which the test was conducted which would be a better representation of practical circumstances. The results of the observations of how much current each subsystem draws and how long the battery will last from a full charge can be seen in table 7.9

Table 7.9: Log for battery life test

System	Current drawn [mA]	Duration of use [hours]
Driving system in use	1100	1.9
Driving system idle	38	55.3
Light	175	12
Camera	200	10.5
Driving system + camera	1300	1.6
All systems in use	1475	1.4

It was found that it took the charger 4 minutes and 26 seconds to charge the battery with 17 mAh. Through calculations it was found that this is equivalent to 15.6 seconds per mAh of charge. Therefore it was estimated that to fully recharge the battery it would take $\frac{(15.6)*2100}{3600} = 9\ hours\ and\ 6\ minutes$.

7.6 Range of Control

7.6.1 Test Procedure for Test 7.6

Aim: To determine the range of operation of the system. i.e how far away the system can be from the operator and still be functional and usable.
Justification: The range of control is an important factor in the field of rescue robotics as the robot will need to be able to be controlled from a safe distance. This requires the robot to be wirelessly operated with the use of a wireless video feed.
Procedure: Testing the range of control can be done in a fairly simple manner. In order to test the range the robot can be turned on and driven away from the operator until a point is reached where the robot ceases to respond to commands from the remote. The distance between the position of the operator and the point where the robot ceased to respond can be measured. This test can be run multiple times just to verify the results and an average can be taken of the distances measured. The distance that the camera feed will operate over will also determine the range that the robot can be operated over. The range of the video feed can be tested in a similar fashion by enabling the video communication and driving the robot away from the receiver until the video feed connection is lost. The distance from the receiver to the point where the robot lost connectivity can be measured.

7.6.2 Results for Test 7.6

The results of the tests to determine the range over which the remote control and video feed communications can operate are as follows:

- Video feed:
  The video feed was found to operate over 35 m after which signal was lost.
  It can be stated that the antennas for the receiver and transmitter were not well matched. The transmitter receiver was circular polarized and the receiver antenna was linear polarised.

- RC control:
  The RC communication was found to operate over 300m after which sight of the robot was lost in the distance and testing became impractical.
  300m was considered to be sufficient result and the test was halted. The test could be continued to find the limit of the range given more time.

7.7 Final Cost of Robot

7.7.1 Test Procedure for Test 7.7

Aim: To determine how much money it cost to research and develop the product as well as to determine how much it would cost to produce the system without the costs of research and development. The cost of the product without research and development is relevant for when the system is to be produced in larger quantities for use in search and rescue applications.
Justification: The project requires a "cost effective robot" in order to meet this specification limits were set. The limit for what could be considered cost effective was 4000 ZAR. The actual cost of the system must now be compared to this goal.
Procedure: The final cost of the system was determined through producing a budget of all the expenses incurred in creating the robot. For the purposes of this project both a research and development budget as well as a final product budget were produced. The research and development budget includes all costs incurred in the design and production including wasted or broken materials as well as failed or broken components. The final product budget includes only the cost of the materials and components that form part of the final product. The remote control and the operators camera feed display are omitted from both budgets as these items are not in harms way and do not need to form part of the "disposable" part of the system.

7.7.2 Results for Test 7.7

The research and development budget can be seen in table 7.10 and the final product budget can be seen in table 7.11 on the following page.

7.7.2.1 Research and Development Product Budget

Table 7.10: Research and development product budget

Item	Units	Cost per unit [ZAR]	Total cost [ZAR]
3mmx40cmx60cm perspex sheet	3	169.01	507.03*
3mmx60cmx60cm Hardboard sheet	2	27.50	55.00
dual H-bridge Chip	1	74.95	74.95
quad MOSFET IC	5	10.73	53.65
STM32f051C6 microcontroller	8	38.00	304.00
Accelerometer	1	153.81	153.81
Mantech Motor	2	142.50	285.00
OrangeRx RC receiver	1	250.00	250.00
Eachine TX01 FPV camera	1	303.38	303.38
Lithium Polymer battery (3.7 V, 2100 mAh)	3	28.43	85.29
Threaded rods nuts and bots	N/A	50.00**	50.00
Total			2122.11

*Donated

**Approximated

7.7.2.2 Final Product Budget

Table 7.11: Final product budget

Item	Units	Cost per unit [ZAR]	Total cost [ZAR]
3mmx40cmx60cm perspex sheet	1	169.01	169.01
3mmx60cmx60cm Hardboard sheet	1	27.50	27.50
dual H-bridge Chip	1	74.95	74.95
quad MOSFET IC	0	10.73	0.00
STM32f051C6 microcontroller	1	38.00	38.00
Accelerometer	1	153.81	153.00
Mantech Motor	2	142.50	285.00
OrangeRx RC receiver	1	250.00	250.00
Eachine TX01 FPV camera	1	303.38	303.38
Lithium Polymer 3.7 V (3.7 V, 2100 mAh)	3	28.43	85.29
Threaded rods nuts and bots	N/A	20.00**	20.00
Total			1406.13

**Approximated

Further tests that would have been carried out given more time can be seen in appendix F.

8 Discussion, Analysis and Test Conclusion

The results presented in chapter 7 will be discussed and analysed in this chapter in order to derive meaning from the results. The discussion will follow the order of the results presented in chapter 7. This chapter includes the discussion of results as well as the comparison of results from tests that were intended to yield comparable results. Graphical means of analysis will be used where appropriate to aid in the visualisation and discussion of the results recorded. Having discussed the results and findings a conclusion for the results of each test will be given.

8.1 Effect of Tire Tread on Acceleration and Maneuverability

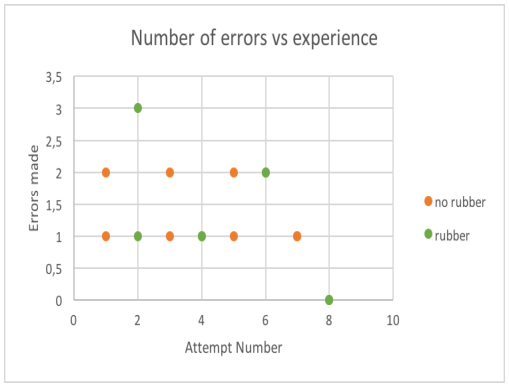
8.1.1 Discussion of Effect of Tire Tread on Acceleration and Maneuverability

This section will discuss the results in section 7.1. The results are based on two separate tests and as such will be discussed in two phases.

Acceleration
The results of the test to determine the effect of rubber wheel coatings on acceleration can be seen in table 7.1 found in section 7.1. These results show that for all cases where the plain perspex wheels were used slipping occurred while for all cases where the rubber coating was applied no slipping occurred. There is an unquestionable difference between using plain perspex wheels and using rubber coated wheels in that the rubber coating drastically reduced slipping while accelerating in a straight line.

Turning
The results of the test to determine the effect of rubber wheel coatings on turning can be seen in tables 7.2 and 7.3 found in section 7.1. These results were taken to compare the effect of rubber on the ability to turn and maneuver but the results were also recorded with respect to the attempt number to show how experience affected the results. The results of this test have been expressed graphically in the form of scatter plots. The scatter plot of the time taken per attempt versus the attempt number and the plot of the number of errors made versus the attempt number can be seen in figures 8.1a and 8.1b on the following page

(a) Scatter plot for time taken per attempt vs attempt number



(b) Scatter plot for errors made per attempt vs attempt number

Figure 8.1: Scatter plots of results of maneuverability test vs experience

Effect of rubber on turning:
It can be seen from the scatter plot in figure 8.1a that the time taken to complete the track was for the most part lower for attempts with rubber on the wheels. It can be seen from the scatter plot in in figure 8.1b that the rubber did not make enough of a difference to the number of errors to be considered a definitive help.

Effect of experience on turning a maneuverability:
In both figure 8.1a and 8.1b it can be seen that less time was taken and fewer mistakes were made as the operator became more experienced. Only 8 attempts were completed and in a short time the operator improved noticeably. The improvement is more apparent with regards to time taken however it can be seen that for the last attempt of each operator not only did they get their best times but they also had no errors on those runs.

8.1.2 Conclusion of Effect of Tire Tread on Acceleration and Maneuverability

The tests explained in section 7.1 are designed to experimentally prove the effects that a rubber coating would have on both the acceleration and maneuverability of the robot. The test also provided insight into the effect of experience on the effectiveness of the operator. The results of the test have been presented and discussed. The conclusions as found through this test are:

Acceleration
The rubber coating on the wheels vastly improved the robots ability to accelerate in a straight line in all tests performed. The use of rubber on the wheels made the robots movements easier to anticipate and control owing to the lack of slipping. The reduction of slipping is also desirable for when the robot is required to climb through the front wheel becoming stuck. It was concluded that rubber coating on the wheels is desirable

and that owing to the definitive results no test to determine the robots ability to climb with or without rubber coated wheels was necessary.

Turning
The rubber coatings effect on the wheels was more complicated than the effect on straight line acceleration. The rubber coating vastly decreased slipping and made the robots movements more predictable making the operation of the robot easier. However the rubber coating decreased the robots ability to make sharp turns it was discovered that this was owing to the design of the robot where wheels cannot rotate freely while coupled to the motor therefore in order to make sharp turns the wheels on one side of the robot must move and the wheels on the other side must slip on the spot this was not possible with the grip from the rubber. It was concluded that the rubber coatings effect on ease of operation outweighed the loss of ability to make sharp turns and as such rubber coatings are recommended.

The conclusions from these two tests resulted in rubber coatings being permanently attached to the wheels to be used for all following tests.

Experience
The tests to determine rubber coated wheels effect on turning were also used to determine the effect of experience on the ability of the operator to operate the robot. It was found that after using the robot for a short period of time the operators ability quickly improved. The effect can be seen over the course of the tests as seen in the scatter plots in figure 8.1. It can also be seen that when switching over to the second track the results were still better than the results from the first track this indicates that the improvement was not merely from practice on the same track but rather from practice and experience of how the robot responds to the controls. It was concluded that the robot was sufficiently simple to operate owing to how quickly an operator can improve and learn to use the system more effectively.

8.2 Comparison of Counter Spin Mechanisms

8.2.1 Discussion of Comparison of Counter spin Mechanisms

The results of the test to determine the effectiveness of the various counter spin mechanisms on stopping the main body from rotating can be seen in section 7.2. The results of the test can be seen graphically in figure 8.2 in the form of a success/failure bar graph. The results portrayed in the bar graph show that with no mechanism the robot was

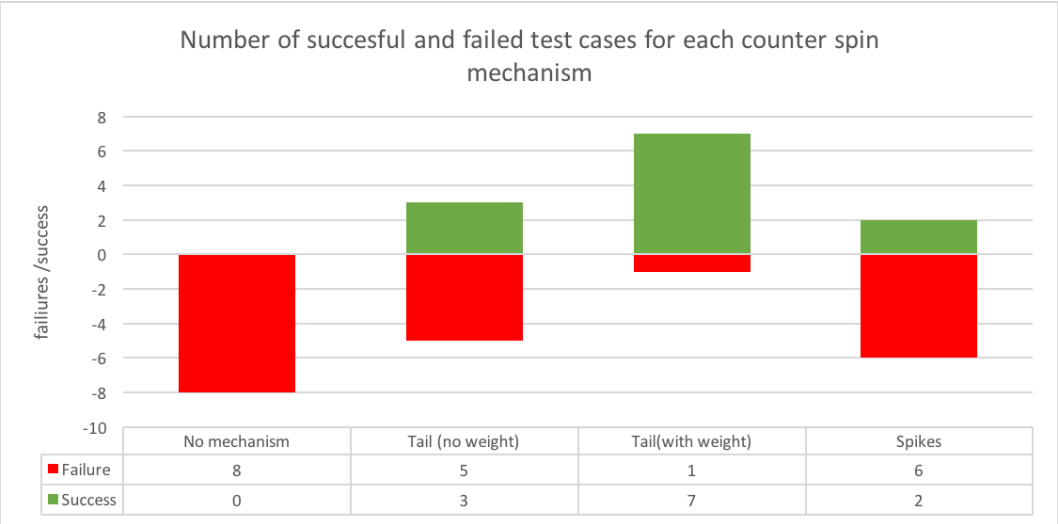


Figure 8.2: Success/failure bar graph of the counter spin mechanism test cases

unable to move in a controlled manner as the main body was spinning. It can also be seen that moderate successes were achieved through the use of the spikes. It was noticed that owing to the torque of the motor the spikes were overcome by the rotational motion of the body for any rapid movement in either direction. Moderate successes were also achieved through the use of the tail with only the batteries as a counter weight, however for a change of direction from forward to backward it can be seen that the tail was not an appropriate solution as the tail would spin over and the controls would be inverted. This was damaging to the tail and to the batteries. The greatest success in the counter spin mechanisms was the tail with an extra weight of 500 grams. With the use of this mechanism it was found that the body only flipped by 180^{o} in the case of the motion changing rapidly from moving forward to moving backward. It was also noted that through careful use of the controls this flip was easily controlled and could be avoided which was not true for any of the other mechanisms.

8.2.2 Conclusion of Comparison of Counter Spin Mechanisms

The tests described in 7.2.1 are designed to experimentally prove the effectiveness of the various counter spin mechanisms designed. The results that have been presented in section 7.2.2 and discussed present information which has been used to draw conclusions

on the effectiveness of the designs and suggest improvements on the designs limitations. The results discussed show that while the spike design helps it is not a suitable solution as it still fails in most cases and leaves the robot highly difficult to control which is the same as for the case of the tail without the added weight. The tail with no added counter weight could be a viable solution if the tail were to be made of a material that is more capable of withstanding shock and if the batteries were housed in a protective casing as shown in the tail design. In order to make the system easier to control with the addition of the tail with no counter weight the controls could be automatically switched through the use of the accelerometer that has be implemented this would create an intuitive control system that changes depending on the orientation of the system. The tail with the addition of the counter weight proved to be a success for the most part with the body only flipping 180º in one test case. However it is a concern that the counter weight will create a difficulty in the wheels hoisting the rest of the robot over an obstacle. It was concluded that the tail without the use of an added counter weight would be the best solution. This conclusion was made as the effect of the counter weight on climbing obstacles could be problematic while the flipping that occurs with this tail is not counter productive to the control of the system if automatic flipping of directions in controls can be implemented.

8.3 Effect of Obstacle Surface on Climbing

8.3.1 Discussion of Effect of Obstacle Surface on climbing

The results of the test to determine the effect that the material of an obstacle will have on the robots ability to overcome it can be seen in section 7.3. In order to determine the effect that the material will have it is important to determine if the wheels can gain enough traction on the surface to begin and carry out the full rotation of the LIM. A graphical representation of the results can be seen in the bar graph in figure 8.3.

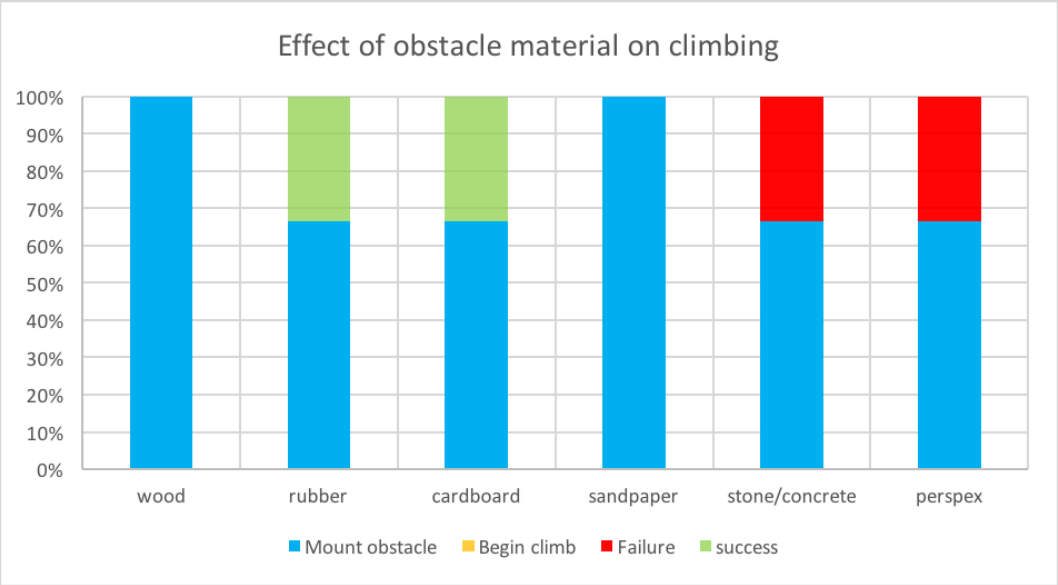


Figure 8.3: Bar graph of outcomes of tests on the effect of the material of an obstacle

The results in the bar graph show that the robot had two failed attempts and two full successes. The remainder of the results show that the robot was able to complete the desired motion and mount the obstacle however it was not capable of pulling itself up. The failed attempts occurred with objects made of stone and perspex, which are very smooth obstacles. It must be noted that a full success and an event where the robot completes the motion but cannot pull itself up are both an indication that the robot was able to gain enough traction to carry out the desired motion.

8.3.2 Conclusion of Effect of Obstacle Surface on climbing

The results of the test to determine the effect that the obstacles surface has on climbing has been discussed. This discussion does not delve into the number of successes that occurred but rather into whether the wheels gained enough traction on the material in order to initiate and carry through the motion until the mechanism had mounted the object. test is simply intended to discover whether sufficient traction can be gained to initiate and complete the flipping motion. Considering only whether the desired motion of flipping was achieved also minimises the limitation of using various size obstacles.

different sized obstacles were used owing to difficulty in procuring obstacles of various materials in a single size.
The results of the test discussed suggest that the rubber coating is not a perfect solution as several failures occurred on the materials with a lower coefficient of friction. The results also show that there was a success with all the materials. This validates that the robot will be able to achieve the desired motion on obstacles of any of these materials if practically deployed, even though it may just take multiple attempts to achieve. while the robot has proven effective on these materials it would be recommended that the wheels be coated with a more suitable material in order to reduce the occurrence of failures which would save time and battery charge in a practical situation.

8.4 Effect of Climbing Motion on Success

8.4.1 Discussion of Effect of Climbing Motion on Success

The results of the test to determine the effect of the climbing motion chosen can be seen in section 7.4 and can be seen reproduced here in table 8.1 for ease of reference.

Table 8.1: Results log for effectiveness of climbing motion

LIM design	Attempt 1	Description	Attempt 2	Description
Version 2.1	2	see ** below	2	see **below
Version3.0	3	see *** below	3	see *** below

The results are enlightening as to the binary success or failure of the motion's ability to climb but do not provide sufficient information. The descriptions given of what occurred can be seen as highly useful information as to the insight of what happened with each motion and how they could be adjusted to produce a working mechanism.

LIM Version 2.1:
Observation: "The rear wheel rotated and mounted the obstacle as desired, the wheel was not able to pull the rest of the body up."

Analysis: The observation of this mechanism provides insight into the problems the mechanism could be facing. It could be reasoned that the problems are caused by the LIM bracket or body beaching on the obstacle, a lack of grip or a lack of torque.

LIM Version 3.0:
Observation: "The front wheel rolled up the face of the obstacle as desired but before mounting the obstacle the LIM continued rotating causing it to then rotate in a complete circle leaving the robot at the starting position having merely swapped the front and rear wheels."

Analysis: The observation of this mechanism provides insight into the problems the mechanism could be facing. Owing to the nature of the mechanism produced, the mechanism will spin the bracket and thus the whole LIM mechanism in the opposite direction to the wheel. This allows the front wheel to roll up the surface. once the wheel reaches the top of the surface the LIM bracket continues to rotate and there is nothing to get the wheel on top of the obstacle. It is noticed that the arm simply rotates over and over again without the ability to mount the obstacle.

8.4.2 Conclusion of Effect of Climbing Motion on Success

Several conclusions can be drawn from the discussion of the effect of the climbing motion. These conclusions relate to the two climbing motions tested and serve to identify the problems and the effectiveness of each. The conclusions can be dealt with for each mechanism individually.

LIM Version 2.1 The problem of being unable to hoist itself up can be owing to beaching, slipping or lack of torque. These problems could be remedied through further research and development, by changing factors such as; the wheel sizes, the grip on the wheels or even the motors. The test results have proven that the mechanism is capable of flipping over onto the obstacle, which confirms the validity of the design. Adjustments or a rear driving force may be able to complete the desired motion, resulting in a working system.

LIM Version 3.0 The problem of not being able to mount the obstacle can be attributed to a lack of forward driving force when the LIMs are rotating this could be remedied by placing driving wheels on the back of the tail for such a situation. This could increase the size of the robot if more than a single wheel is attached and could affect maneuverability. This leaves the design less suitable to the application for this robot as adding further drive systems to the back would require more design and implementation not suitable for the timeline. However, if implemented from the start, with the information presented in this project this could be attempted as a valid design with the use of a rear driving system.

The information provided through testing has shown that either mechanism could hold promise. The direction that further research and development should take in order to move towards producing a working prototype of either mechanism has been shown as well.

8.5 Duration of Battery Life

8.5.1 Discussion of Duration of Battery Life

The results of the the test to determine the expected battery life of the system can be seen in section 7.5. The battery duration results can be seen represented graphically in the form of a bar graph shown in figure 8.4. It can be seen from these results that

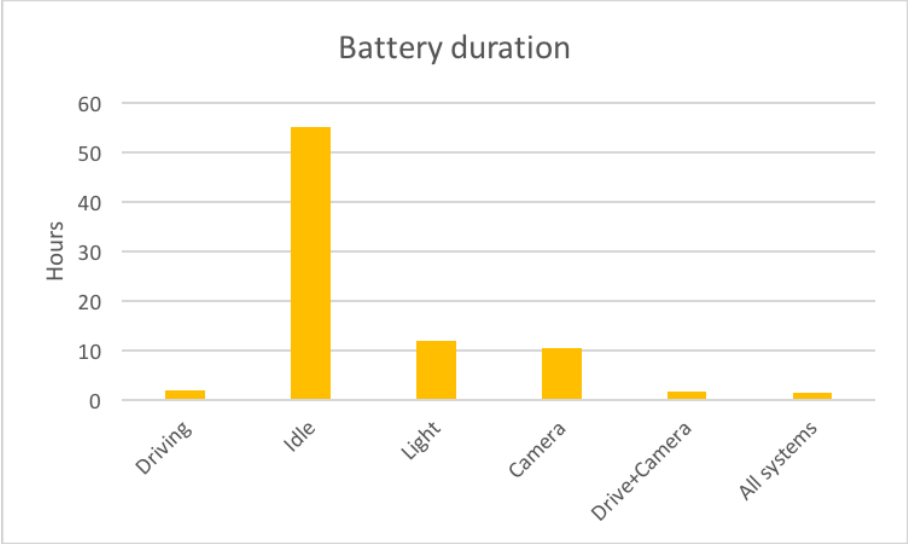


Figure 8.4: Bar graph of battery durations in different operating states

in the idle state the battery will last much longer than other states. The results are shown graphically without the idle result in figure 8.5 in order to be able to see a the other results on a more suitable scale for comparison. It can be seen that the driving

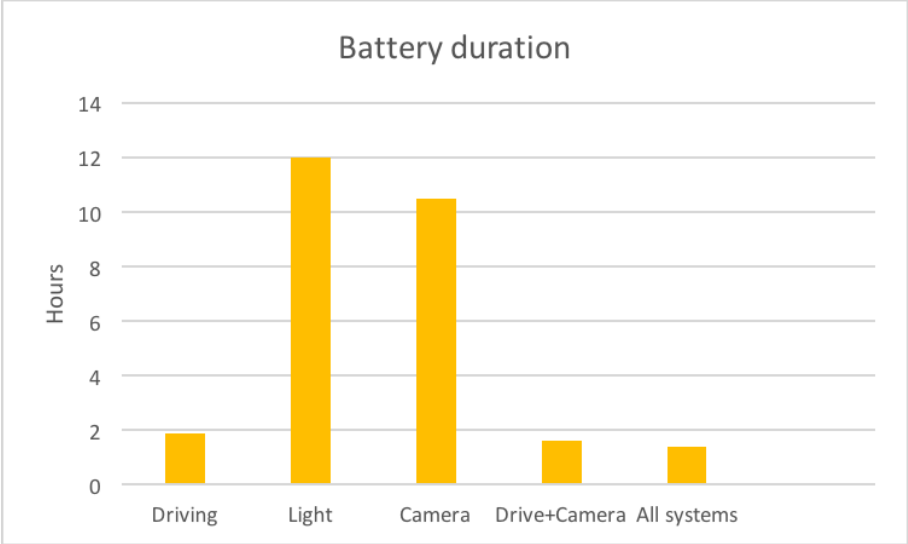


Figure 8.5: Bar graph of battery durations in various active operating states

mechanism consumes the most power. It can be assumed that driving whilst using the camera is the most common state for the system to be in, thus the general expected battery life will be 1 hour and 36 minutes which will be slightly lowered if the light is being used intermittently. With all systems running (the robot moving the camera on and the light on at 75% duty cycle) the battery is expected to last an hour and 24 minutes. Since the recharge time of the battery was found as 9 hours and 6 minutes, 7 sets of batteries would be needed in order to operate the robot continuously while batteries are recharging. (This is under the assumption that the robot is operated with all systems on and the battery would last 1.4 h)

8.5.2 Conclusion of Duration of Battery Life

The battery life has been discussed and the conclusions are as follows. The battery life is long enough to be able to enter an urban search and rescue environment and have sufficient time to search the environment before needing to return to the operator. The recharge time is significantly longer than the time that the battery would last for but "down time" could be avoided by having multiple battery packs charging at any time. Having multiple battery packs charging would not increase the cost of the equipment that is sent into the field as if the robot is lost to the environment the batteries would be with the operator and would not be lost in the field.

8.6 Range of Control

8.6.1 Discussion of Range of Control

The results of the test to determine the range of control offered by the system can be seen in section 7.6.
The range of control offered by each communication device as shown in the tests are :

- Video: 35 m
- Radio Control: 300+ m

The antennas used for testing the video feed were noted to be mismatched. The result of this mismatch would result in a -3dB loss in signal. A -3dB loss in signal is equivalent to a loss of about half the strength of the signal. This loss does not translate into halving the range but it can be said that without this -3dB loss the range would increase. The range limit of the radio control could not be found and further testing would need to be done to find the limit of the range.

8.6.2 Conclusion of Range of Control

The range of control of the remote control system is deemed sufficient for general USAR needs. The camera feed range is seen as a severely limiting factor on the range however

the video feed serves as proof of concept that an FPV camera can be placed in a rescue robot for use by an operator. The camera feed range could be increased in the future with the use of well matched antennas and a higher power transmitter.

8.7 Final Cost of Robot

8.7.1 Discussion of Final Cost of Robot

The final cost of the project was calculated by adding up the cost of all the materials and components used as shown in section 7.7. The cost totals as well as the budgets can be seen in figure 8.6.

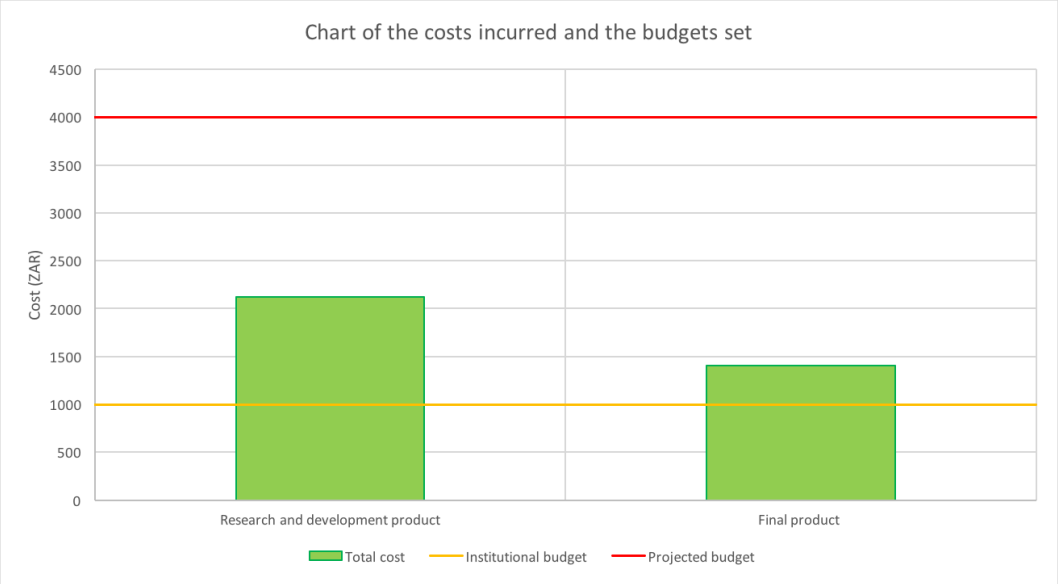


Figure 8.6: Bar graph of costs to develop product and the budgets set

The costs are calculated in two separate formats (Research and development costs and final product costs) which can be discussed separately keeping in mind that the budget from the university is 1000 ZAR and the limit that was set on the system being considered disposable is 4000 ZAR.

8.7.1.1 Research and Development Product Budget

The research and development project budget involves calculating the costs of all materials and components used throughout the course of the project. This budget includes waste from unused materials as well as failed or broken attempts. This budget is described in section 7.7.2.1. In the graph it can be seen that this budget exceeded the budget imposed by the university by 1122,11 ZAR however it slightly above half the budget that is used as a metric to determine if the robot is cost effective.
The split of the research and developments costs incurred can be seen in figure 8.7.

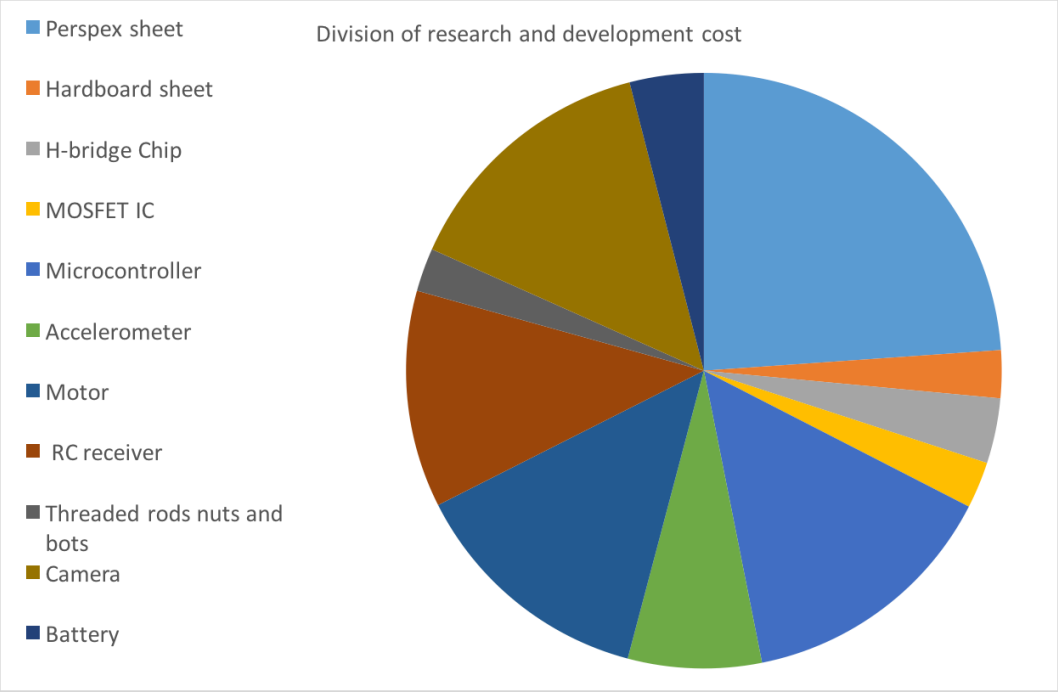


Figure 8.7: Pie chart of the division of costs incurred during research and development.

The pie chart shown in figure 8.7 indicates that the most significant contribution towards cost came from perspex, this was owing to the various iterations of the LIM designs and replacing the tail which would not be a cost to be incurred in production stages after research and development.

8.7.1.2 Final Product Budget

The final product budget involves calculating the cost of all materials and components that are included in the final build of the product. This budget does not include failed attempts and waste as it is a closer representation to the cost that can be achieved for each robot produced once research and development is completed. The calculation of this budget can be seen in section 7.7.2.2. This budget slightly exceeds the budget imposed by the university (by 406,13 ZAR). However, it is just over a quarter of the budget used as a metric to determine whether it is a cost effective system.
The split of the costs incurred in creating only the final product can be seen in figure 8.8.

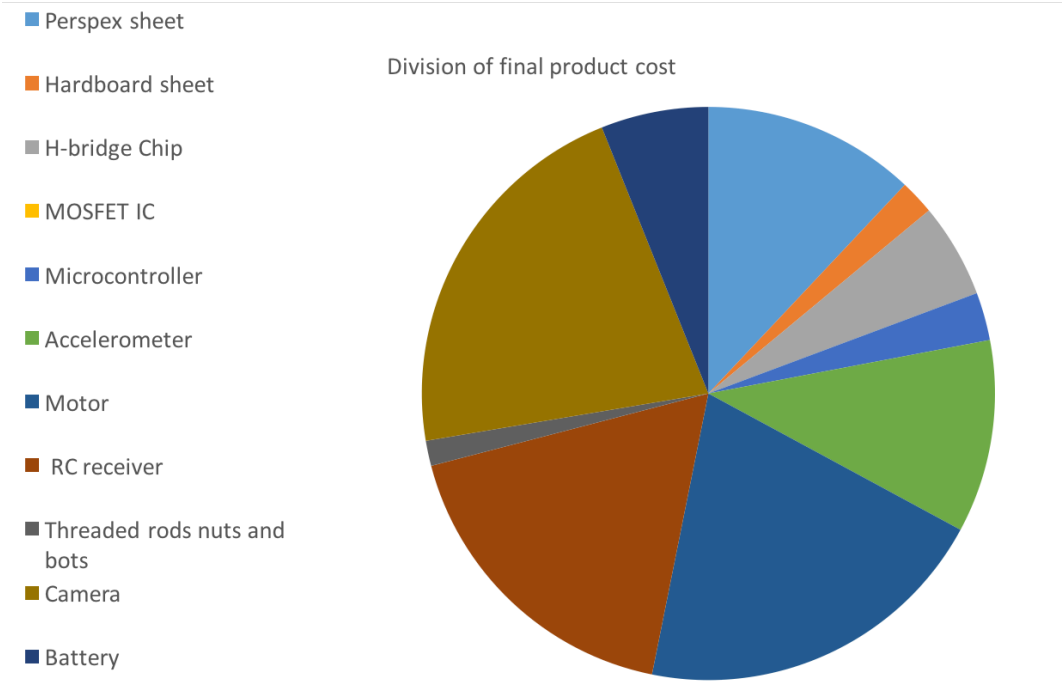


Figure 8.8: Pie chart of the division of costs incurred to produce only the final product.

Figure 8.8 shows that the main costs incurred in producing the final product are the camera, the RC receiver and the motors. This is acceptable as these cost are necessary to produce a tele-operated observation robot. It can also be seen that the cost of perpex has been decreased without the need for testing various LIMs.

8.7.2 Conclusion of Final Cost of Robot

The cost expected to produce the final product after the research and development phase is significantly below what was set as a metric for cost effectiveness and being considered disposable. For this reason the final product is considered successful in being cost effective and disposable if lost. As a result many of the aspects that were

compromised owing to trying to keep the budget down can be revisited and improved upon making the system more robust and useful while keeping the system cost effective.

9 Project Conclusions

This chapter will draw conclusions from the findings and results presented in chapter 7 and the discussion and analysis undergone in chapter 8. These conclusions will be aimed at defining whether the design decisions made in the process of creating the system were effective or not as well as dealing with why they were or were not effective. This chapter will discuss an overall system conclusion based on whether it met the specifications defined in chapter 3 and will then go on to present notable limitations and recommendations.

9.1 Satisfaction of Specifications

The specifications listed in chapter 3 serve as a guideline for the objectives to be achieved during the design and implementation phases. The success or failure to achieve these specifications is discussed in this section.

9.1.1 Size

Fulfillment of the size specification was achieved through checking whether the dimensions of a rectangular box that could contain the robot was within the size constraint. It was found that the containing box measured:

- Height: 125mm
- Width: 235mm
- Length: 330mm

These dimensions are further reduced by removing the detachable LIMs for storage and transportation. This was within the initial size specification. As such the specification is considered fulfilled.

9.1.2 Agility

The system did succeed in performing the desired motion of approaching an object letting the front wheel tough the object and become locked from friction with the object after which the LIM rotates the rear wheel over the front wheel and on to the object. This is considered a successful proof of concept that the mechanism would work if the build of the LIMs were more robust. The proof of concept was achieved on a 140mm high

obstacle. Therefore the specification was not fully met however the concept was proven and improvements can be made.

9.1.3 Cost

Determining whether the project was within the specified budget can be determined through the results of the cost analysis completed through sections 7.7, and graphically displayed in figure 8.6. The bar graph contains all the information relevant in determining whether the specification was fulfilled. The institution budget imposed was not fulfilled however the budget proposed for the system to be considered cost effective and disposable if lost was fulfilled. The specification of producing a robot cost effective enough to be considered disposable if lost in the field is considered fulfilled.

9.1.4 Vision

The criteria of having a wireless means of video communication in the robot was achieved through the use of FPV camera and NTSC transmission. The video link set up shows proof of concept in that it is possible to place a camera in the robot. The range of the video link was only 35 m. however this can be improved upon with well matched antennas and a higher power transmitter. This specification was deemed satisfied.

9.1.5 Tele-operation

Satisfaction of this criteria is fulfilled in two parts.

1. The system satisfies the requirement to be tele-operated without the use of a physical tether as it operates (in the correct manner) though the use of an RC transmitter and receiver.

2. The communication operates over a range of 300m as shown in section 7.6.

These two points confirm that robot meets the specification listed under tele-communication.

9.1.6 Battery Life

The battery life of the system exceeds the goal duration of 1 hour. Therefore the battery duration specification is deemed fulfilled. The ability to keep the system constantly going by replacing the batteries required multiple batteries however this is not a significant issue owing to the spare batteries not being a cost that needs to be written off if the system is lost in the field. The battery life specification was deemed satisfied with the requirement of needing spare batteries.

9.1.7 Robustness

The criteria listed under robustness has no definite means of testing. The ability to fulfill this specification was assessed through the continued operation of the system

throughout testing. The main body of the robot proved to be highly durable and did not yield under the pressures of testing. The LIMs and wheels did not satisfy this specification. The LIMs often came loose owing to their easily detachable nature, this left the LIMs loose and off centre making climbing obstacles difficult. The nuts coupling the wheels to the axles often came loose leaving the wheels standing still while the gears were spinning. These issues would not be acceptable in practical operating conditions and would need to be solved.

9.1.8 Simplicity of Operation

Assessing whether the operation is simple is not a binary outcome. It can be noted that the controller set up in order to drive the robot is significantly more simple than those researched, owing to a more intuitive approach and fewer controller movements needed for operation. It can be considered that the simplicity of operation presented has improved upon previous attempts. This simplification was deemed to be a step in the right direction and the specification is considered fulfilled.

9.2 Limitations

During the course of the project several limitations were encountered. These limitations have had a noticeable effect on the project and the final work delivered. This section deals with various limitations that were met and how they affected the project.

Laser cutter

- Access:
  The laser cutter was only accessible from 8 AM to 10 AM Monday to Friday. This very limited slot in which to laser cut led to many delays and time wasted without the mechanical components needed.
- Accuracy:
  The laser cutter used had an inherent 0.2 mm inaccuracy owing to the width of the laser, this could be accounted for for the most part by adjusting measurements in designs. The laser cutter experienced problems with cutting circles. The result was an off centre tapered cylinder rather than a centered cylinder when cutting holes. This resulted in many issues with the gears and wheels as nothing had a tight straight fit onto the axles and shafts.

Budget

The budget allowed from UCT was below the budget that would have satisfied the cost effective specification given. This resulted in the need for costs to be kept to a minimum and for donors of certain materials to be found.

Timeline
The project timeline was intended to be 12 weeks, owing to unforeseen circumstances the project topic was administered two weeks into the timeline, this resulted in a total time of 10 weeks to complete the project.

Access to Components

- Access:
  Components from the UCT electronics store were available from 8AM to 3:30 PM Monday to Friday this resulted time spent in the evenings without the components needed to complete certain circuits.
- Availability:
  Certain components where not available at the UCT electronics store and as such needed to be ordered. This resulted in waiting times of over a week for components such as the quad MOSFET package.

Tracking Variables in Atollic
Atollic is capable of tracking variables when the microcontroller is running connected to a PC. This allows for investigation of the values being read from peripheral devices such as the accelerometer and the RC receiver. The free version of Atollic only allows for tracking of these variables while the program is halted. This resulted in the use of rough estimates of expected values for certain events and states measured from the peripheral devices as they could not be tracked in real-time.

Protest Action Protest action that resulted in closing campus had a large effect on the work done in the project. The protest action occurred during the final few weeks of the project which were crucial time left for the completion of certain tasks. These tasks could not be completed with the interrupted access to campus facilities. The psychological effect of disruptions at the end of a long year and a long project procedure broke much momentum and resulted in the abandonment of certain tasks in order to finish in time. These abandoned tasks could not simply be picked up again and completed with the extra days given. The protest had more of an effect than the loss of time and it can be noted that a day lost to protests is not equal to a day added on to the deadline of a project.

9.3 Recommendations

Failure to achieve specifications need not be considered a design failure. The design carried out in this project has highlighted downfalls in certain designs and the implementation of these designs. The failure of components to achieve the desired effect has resulted in a learning experience. This highlights aspects that must be further improved upon. These aspects can be achieved through changing design decisions based on the information presented in this project. This section deals with any recommendations found from the experience gained in completing this project.

LIMs and Wheels
The LIMs and wheels proved effective enough to meet the criteria in intermittent phases between fixing and rebuilding. This had a large impact on the effectiveness of the design as it was not robust enough. The use of threaded rods which required nuts to be tightened in order to couple gears and wheels to the shafts was also not ideal owing to the loosening of the nuts. These problems can be fixed through the fabrication of a more robust LIM bracket, similar to the one in Matthew Wilson's project [22]. One further recommendation would be to couple the LIM to the motor shaft through a more robust means than simply attaching a tight gear to the motor shaft.

H-bridge
The H-bridge designed, but not completed, would be a large help in the implementation of motors with a higher voltage rating and current draw. This could prove useful to stop the motors from stalling in certain circumstances.

Gear Ratio
The gear ratio requirements for the LIM impose stringent requirements on the fact that the gear ratio in the LIM must be a torque reduction gear ratio. This leads to the robot being too fast for easy control and not powerful enough in certain circumstances. This issue can be circumvented through the use of a speed reduction gearbox placed before the shaft which is connected to the gears. This allows for an increase in torque through the primary gearbox which will bear no effect on the desired operation of the LIM.

Rear Driving Wheel
The implementation of a rear driving wheel with the motion achieved through LIM2.1 or LIM3.0 could prove useful in assisting the system to overcome obstacles once the wheels have mounted the obstacle. This could be implemented with ease at the back of the tail with no need for speed control. This wheel could simply be turned on when needed to overcome obstacles and turned off to free wheel when not needed. Similarly, in order to assist the tail in getting over the obstacle, a linear shaft can be extended in order to lift the tail to the height desired. A linear actuator could be considered, however, a simple gear mechanism with the use of a rack and pinion could be implemented.

Non-circular Wheels
The use of wheels with cut outs on the outside circumference could assist in the gripping and climbing of step shaped obstacles. These wheels could be shaped in order to mesh with the step in order to help the wheel overcome the step once the wheel has mounted the obstacle.

Investigation of Better Weight Distribution
The weight of the batteries on the tail resulted in the robot having difficulty in pulling the tail up steps. A better weight distribution with the use of the tail could improve the robots ability to deal with this problem.

Size Varying LIMs
Size varying LIMs could also be investigated through the use of a pulley system rather than gears in the LIM. The pulley system could make use of elastic belts and a rack and pinion mechanism to extend and contract the LIM. This solution would be complicated to design and fabricate but could prove useful in climbing obstacles of various sizes if realised.

Literature

[1] N. G. Hockstein, C. G. Gourin, R. A. Faust, and D. J. Terris, "A history of robots: From science fiction to surgical robotics", *Journal of Robotic Surgery*, vol. 1, no. 2, pp. 113–118, April 2007. DOI: 10.1007/s11701-007-0021-2.

[2] J. Capek, *Opilec*, ser. Lelio A Pro DelWna. Aventinum, 1925.

[3] k. Capek, *R.u.r. (rossum's universal robots)*. Penguin Group, 2004.

[4] I. Asimov, *I, robot*. Harper Voyager, 2013.

[5] B. Rocco, G. Albo, and R. F. Coelho, "From leonardo to da vinci: The history of robot-assisted surgery in urology", *BJU International*, vol. 108, no. 11, pp. 1714–1714, 2011. DOI: 10.1111/j.1464-410x.2011.10600.x.

[6] A. Davids, "Urban search and rescue robots: From tragedy to technology", *IEEE Intelligent Systems*, vol. 17, no. 2, pp. 81–83, March 2002. DOI: 10.1109/MIS.2002.999224.

[7] A. Jacoff, E. Messina, B. A. Weiss, S. Tadokoro, and Y. Nakagawa, "Test arenas and performance metrics for urban search and rescue robots", in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 4, October 2003, 3396–3403 vol.3. DOI: 10.1109/IROS.2003.1249681.

[8] D. S. Ignatova and R. K. Oransky, "Mechatronical approach to investigations of rescue operations", in *14th International Conference Mechatronika*, Jun. 2011, pp. 103–108. DOI: 10.1109/MECHATRON.2011.5961098.

[9] B. Hall, *Lessons from first responders: Sometimes you shouldn't be a hero.* May 2013. [Online]. Available: http://www.slate.com/articles/health_and_science/science/2013/05/rescuers_turning_into_victims_lessons_from_first_responders_on_saving_people.html.

[10] *United states department of labor*. [Online]. Available: https://www.osha.gov/pls/oshaweb/owadisp.show_document?p_table=PREAMBLES&p_id=839.

[11] C. R. Stein, S. Wallenstein, M. Shapiro, D. Hashim, J. M. Moline, I. Udasin, M. A. Crane, B. J. Luft, R. G. Lucchini, W. L. Holden, and et al., "Mortality among world trade center rescue and recovery workers, 2002-2011", *American Journal of Industrial Medicine*, vol. 59, no. 2, pp. 87–95, April 2016. DOI: 10.1002/ajim.22558.

[12] *Urban search & rescue*. [Online]. Available: https://www.fema.gov/urban-search-rescue.

[13] A. Meystel, E. Messina, N. I. of Standards, and T. (U.S.), *Measuring the performance and intelligence of systems: Proceedings of the 2000 permis workshop, august 14-16, 2000*, ser. NIST Special publication. The National Institute of Standards and Technology, 2001, pp. 253–259. [Online]. Available: https://books.google.co.za/books?id=yAhRAAAAMAAJ.

[14] I. C. Council, *International building code 2006*, ser. INTERNATIONAL BUILDING CODE. International Code Council, 2006, p. 212. [Online]. Available: https://books.google.co.za/books?id=7TzonAAACAAJ.

[15] J. Wong, C. Robinson, *et al.*, "Urban search and rescue technology needs: Identification of needs", *Federal Emergency Management Agency (FEMA) and the National Institute of Justice (NIJ). Document*, vol. 207771, 2004.

[16] M. Takahaashi, K. Yoneda, and S. Hirose, "Rough terrain locomotion of a leg-wheel hybrid quadruped robot", in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 1090–1095. DOI: 10.1109/ROBOT.2006.1641855.

[17] J. A. Smith, I. Sharf, and M. Trentini, "Paw: A hybrid wheeled-leg robot", in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 4043–4048. DOI: 10.1109/ROBOT.2006.1642323.

[18] S. Hirose and H. Takeuchi, "Study on roller-walk (basic characteristics and its control)", in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, April 1996, 3265–3270 vol.4. DOI: 10.1109/ROBOT.1996.509210.

[19] N. Eiji and N. Sei, "Leg-wheel robot: A futuristic mobile platform for forestry industry", in *Proceedings of 1993 IEEE/Tsukuba International Workshop on Advanced Robotics*, November 1993, pp. 109–112. DOI: 10.1109/ICAR.1993.337208.

[20] *Turtlebot3 official wiki*. [Online]. Available: http://turtlebot3.robotis.com/en/latest/.

[21] T. J. Mathew, "Scarab: Development of a rugged, low cost, inspection-class robotic platform", Master's thesis, 2015.

[22] M. Wilson, *Development of a low-cost, mid-sized, tele-operated, wheeled robot for rescue reconnaissance*, 2013.

[23] D. Redell, "Newsletter of the livermore unit of the national association of rocketry", January 1998.

[24] Gzip and Instructables, *H-bridge on a breadboard*, May 2016. [Online]. Available: http://www.instructables.com/id/H-Bridge-on-a-Breadboard/.

[25] A. WILLIAMS, *Microcontroller projects using the basic stamp*. CRC PRESS, 2017.

[26] *Projectw2011*. [Online]. Available: http://hades.mech.northwestern.edu/images/8/84/ProjectW2011.pdf.

[27] *Rc interface (dxi/dxo) - 2.007 arduino nano carrier*. [Online]. Available: https://sites.google.com/site/2007arduino/hardware-description/rc-interface-dxi-dxo.

[28] *Understanding rc servos*. [Online]. Available: http://www.rchelicopterfun.com/rc-servos.html.

[29] *Ppm-encoding*. [Online]. Available: http://www.mftech.de/ppm_en.htm.

[30] DoYeouKu, *Sensor-ku*, Sep. 2017. [Online]. Available: https://github.com/DoYeouKu/Sensor_Ku/tree/master/modules.

Appendices

A [Formal project outline]

ID:	**JCP17-01**
SUPERVISOR:	J.C Pead
TITLE:	**A cost effective, tele-operated observation search and rescue robot.**
DESCRIPTION:	Disaster environments are dirty and dangerous, which makes the role of rescue aid ideal for a robot. The main role of a rescue robot is to provide beneficial information to the rescuers without burdening the crew with regard to environment difficulties or concern of the wellbeing of their device. Larger robots could struggle to navigate the terrain or can be costly to purchase or maintain. Small low-cost platforms often do not have these capabilities to survive the environment and provide no value to a rescue team. Alternative approaches were tested (T. J. Mathew, 2015; M.Wilson, 2013) and provided plausible systems to the costly larger robots. This project will investigate a hybrid design between the earlier projects conducted at UCT. The focus of this project will be on overcoming obstacles that may be found in a USAR environment with a strong focus on off the shelf and cost effective solutions to reduce concerns over vehicles in disaster zones.
DELIVERABLES:	**Objectives:** The main objectives of this project are: i. Understand the requirements of the project ii. Conduct a literature review of previous work in this field and critically evaluate current technology/research iii. Design a "small cost effective remote observation vehicle" capable of manoeuvring though an urban building environment iv. Test and compare vehicle capabilities with some common obstacles **Deliverables and expectations:** i. A literature review report (Hand-in 2 weeks into the project) ii. Design and construction of a mobile remotely operated vehicle iii. Weekly attendance of individual meeting with supervisor iv. A project report v. Satisfactory completion of all ECSA ELOs vi. Departmental seminar at the end of the project vii. Poster and demonstration at the Departmental Open day
SKILLS/REQUIREMENTS:	Practical student, Comfortable with electronics, dynamics, programming
ELO3: Engineering Design Perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes.	This project involves specification, design, prototyping and testing of the small hybrid platform attempting to combine the advantages of both platforms in the earlier projects.
EXTRA INFORMATION:	T. J. Mathew, "Scarab: Development of a rugged, low cost, inspection-class robotic platform", MSc dissertation, University of Cape Town 2015 M.Wilson, "Development of a low-cost, mid-sized, tele-operated, wheeled robot for rescue reconnaissance", Final year project, University of Cape Town, 2013

B [Quote from Inuktun]

Gmail **Jordan Haskel <jordhask@gmail.com>**

RE: Request for Quote (RFQ) for Inuktun Services Ltd via AZoRobotics.com (ID 1218)

Bis Pradhan <bpradhan@inuktun.com> Sat, Sep 2, 2017 at 1:21 AM
To: "jordhask@gmail.com" <jordhask@gmail.com>

Hi Jordon,

Thank you for the enquiry. Unfortunately, we don't manufacture any rescue robots. Our inspection robot price starts from USD $35,000.

Kindly go through our website http://inuktun.com/en/ to find out our variety of robotic crawlers we manufacture. But still, if you want us to customize and build a rescue robot for you then that would probably cost you over USD $100,000.

Kindly share your thoughts on the same. Many thanks.

Regards,

Bis

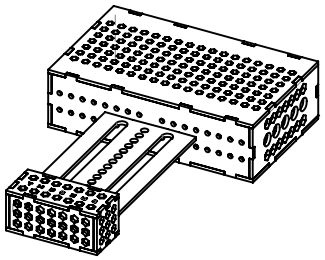
From: RFQs@AZoRobotics.com [mailto:RFQs@AZoRobotics.com]
Sent: Thursday, August 31, 2017 3:39 PM
To: Inuktun Info; Inuktun Sales
Cc: Rolf Easto
Subject: Request for Quote (RFQ) for Inuktun Services Ltd via AZoRobotics.com (ID 1218)
Importance: High

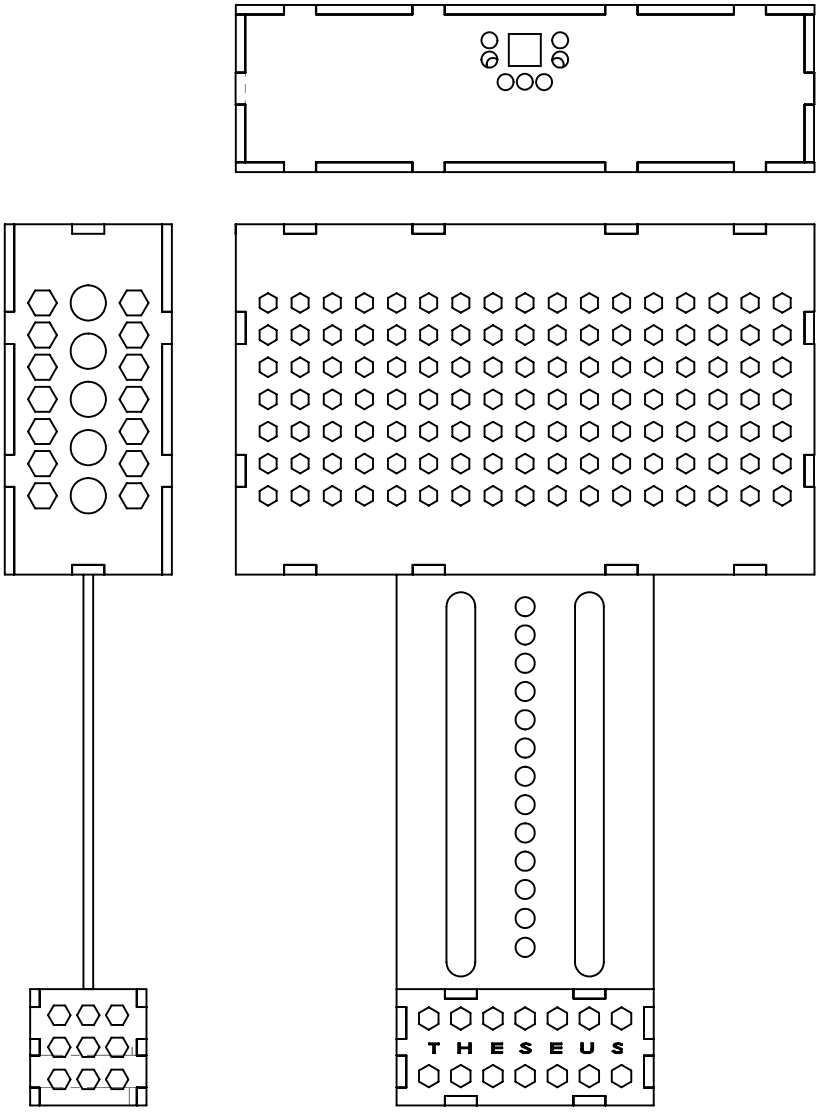
AZO NETWORK

RFQ Submission

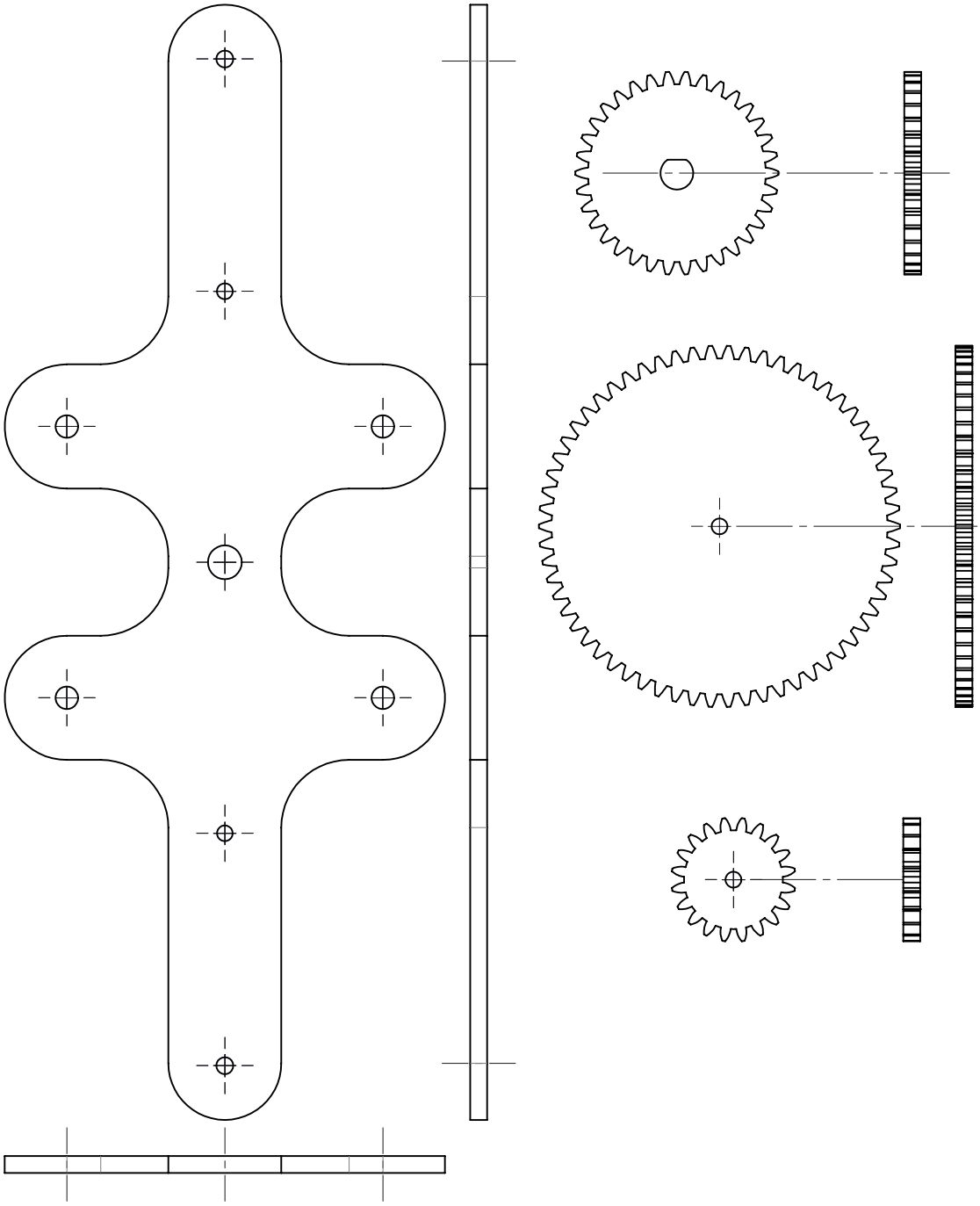
C [Solidworks schematics]

The Solidworks schematics shown in this appendix were all designed with the intention of fabricating the parts using a laser cutter. All parts were designed to be cut from 3mm thick hardboard or perspex.

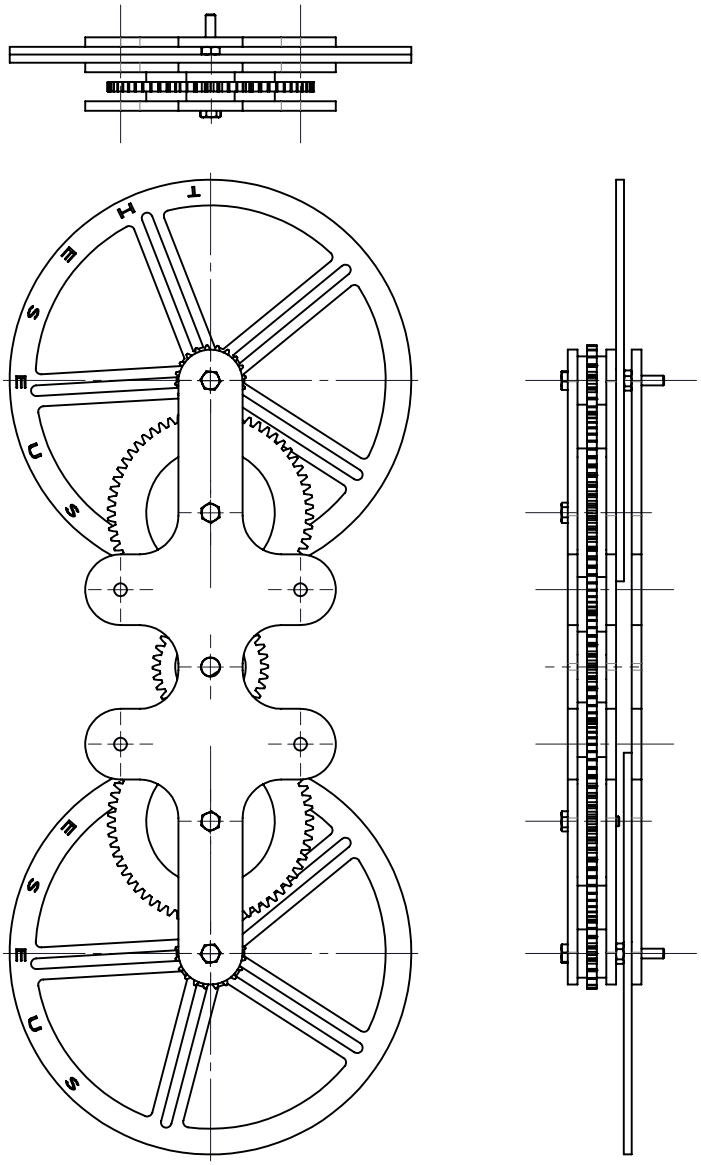




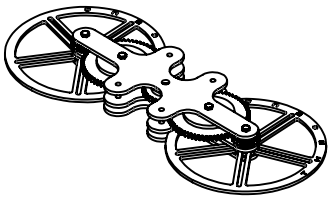
Scale: 1:2 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: Theseus build with tail		
Checked :	Material : Wood and Perspex	Drawing Number :	Rev. : A	Sheet : 1 of 1

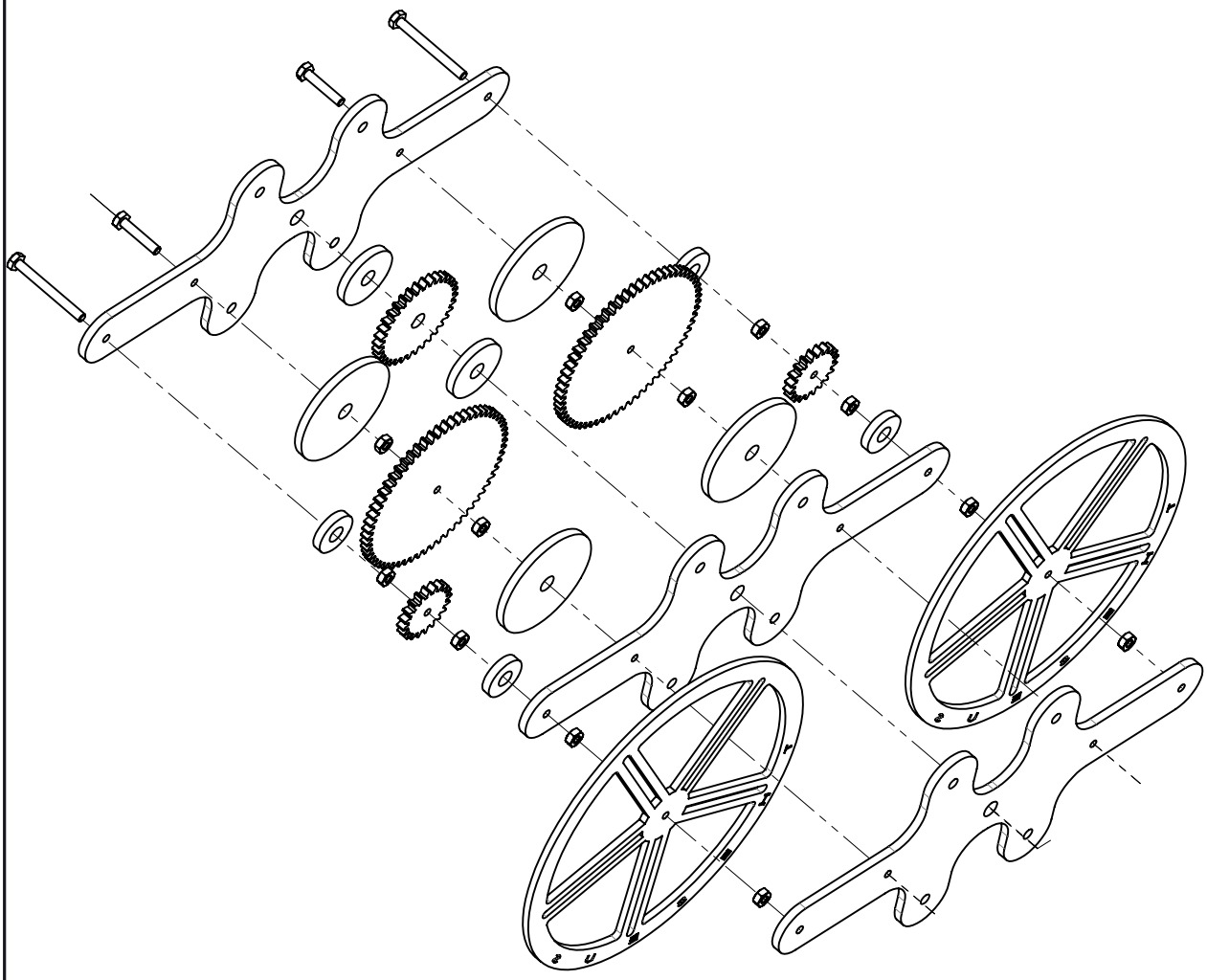


Scale: 1:1 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title:Gears and bracket		
Checked :	Material : Perspex	Drawing Number :	Rev. : A	Sheet : 1 of 1

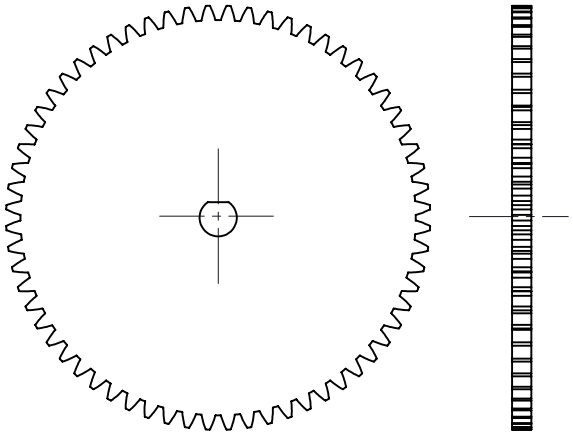


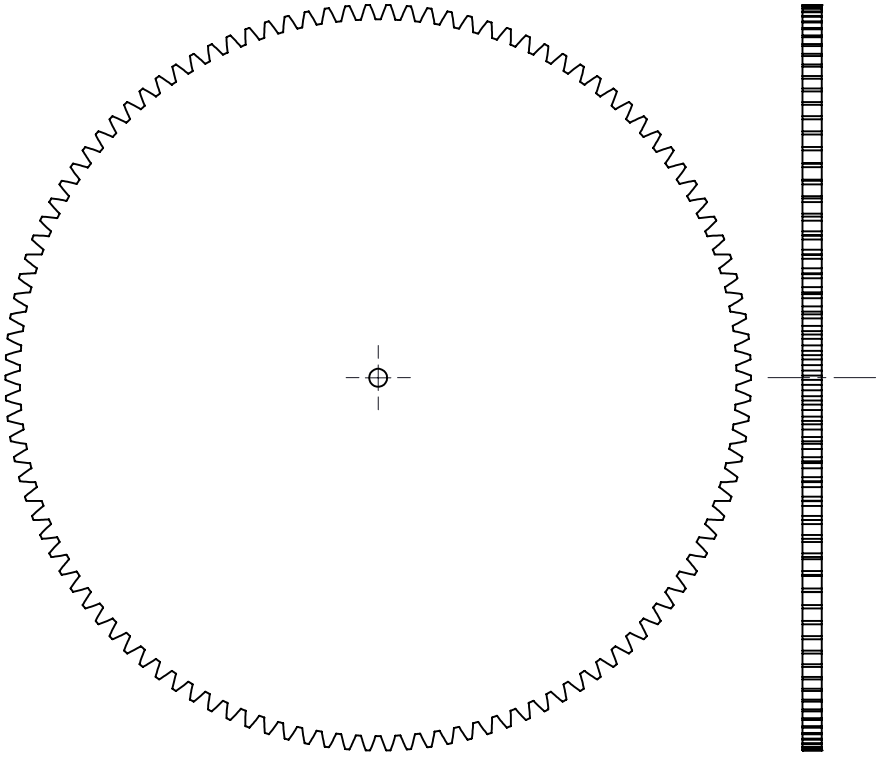
Scale: 1:2 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title:LIM2.1		
Checked :	Material : Perspex and Wood	Drawing Number :	Rev. : A	Sheet : 1 of 1



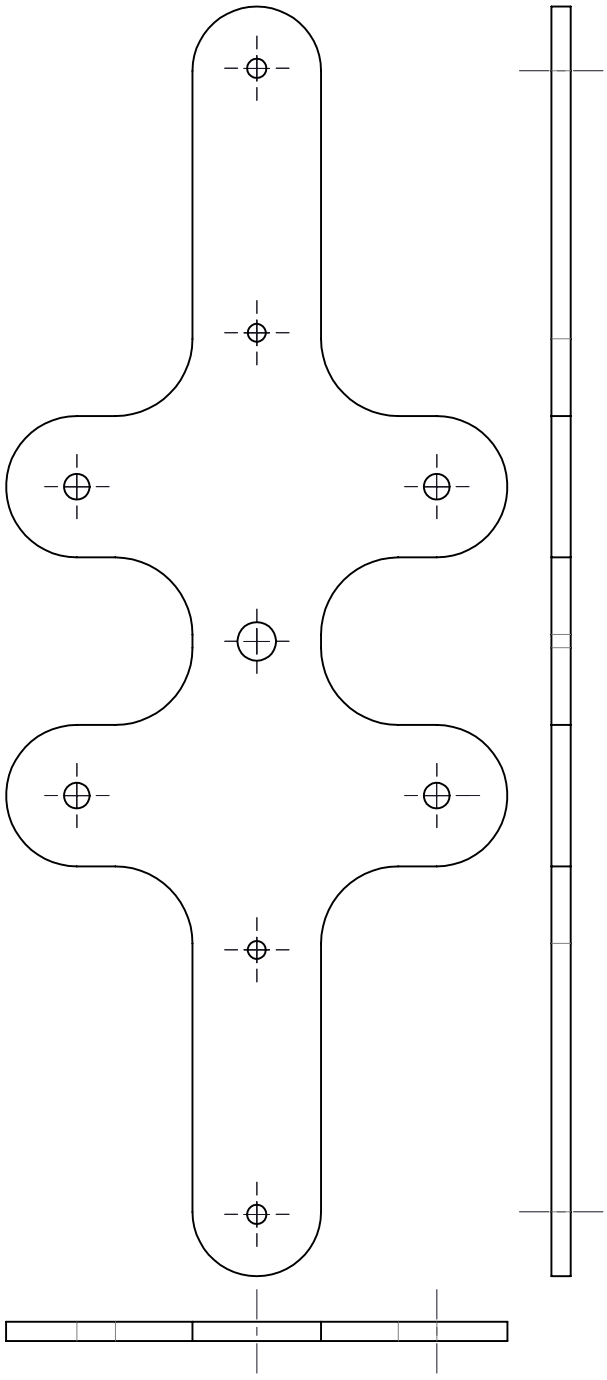


Scale: 1:2 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: LIM2.1		
Checked :	Material :Perspex and Wood	Drawing Number :	Rev. : A	Sheet : 1 of 1

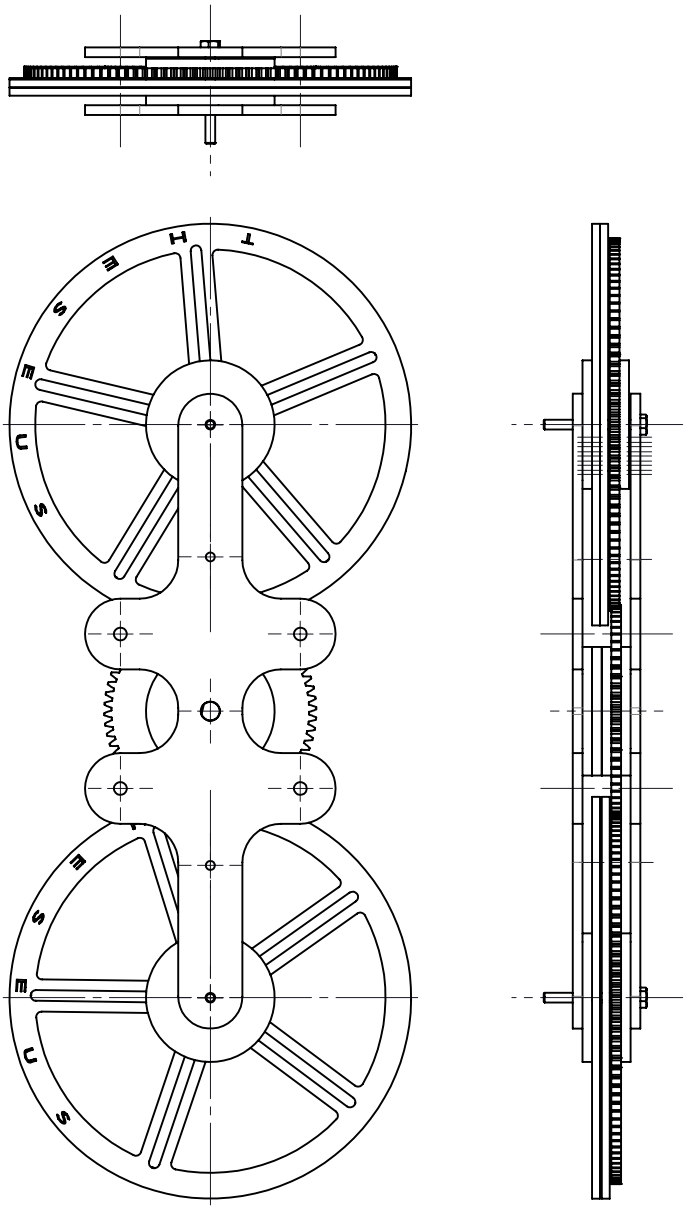




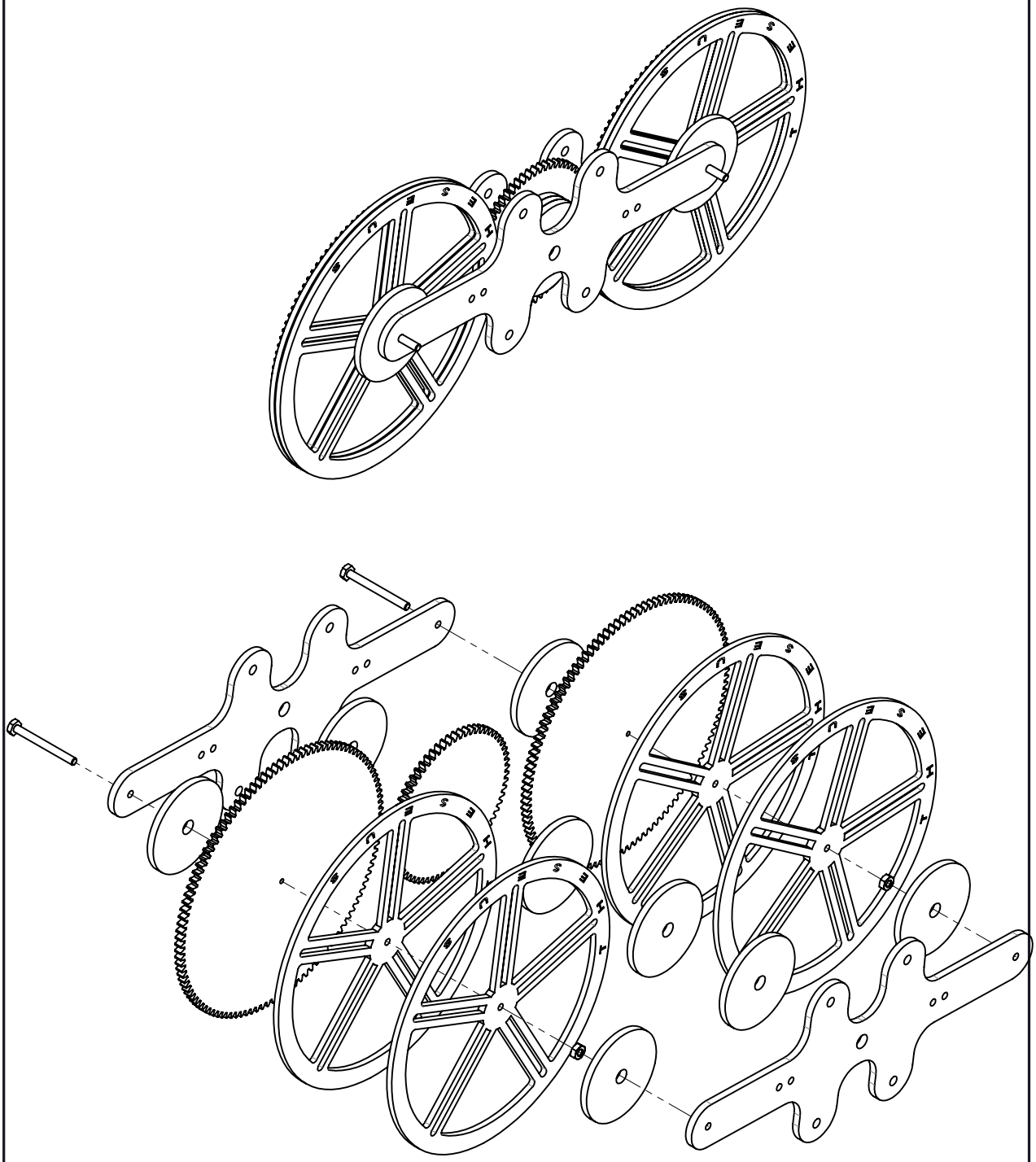
Scale: 1:1 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: Gears 3		
Checked :	Material : Perspex	Drawing Number :	Rev. : A	Sheet : 1 of 1



Scale: 1:1 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: LIM holder for ver3.0		
Checked :	Material : Perspex	Drawing Number :	Rev. : A	Sheet : 1 of 1



Scale: 1:2 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: LIM3.0 views		
Checked :	Material :Perspex and wood	Drawing Number :	Rev. : A	Sheet : 1 of 1



Scale: 1:2 on A4	University of Cape Town Department of Mechanical Engineering			
Drawn By: Jordan Haskel	All un-toleranced dimensions to adhere to ISO 2768-m	Title: Internal wheel lim design		
Checked :	Material : Perspex	Drawing Number :	Rev. : A	Sheet : 1 of 1

D [Logisim simulation]

Logisim was used to simulate the anti shoot through circuit described in section 6.2.1.4.The circuit consists of one NOT gate and two NAND gates. The MOSFETs from the H-bridge are also included in the simulation along with the BJT transistors. The BJT transistors are replaced by NOT gates which perform the logic inversion that occurs through the BJT transistors in the switching circuit. The anti shoot-through circuit has two input lines:

- Direction line can be high (3.3v) or low (0v)
- PWM line switches from high (3.3v) to low (0v) with a specified frequency at variable duty cycle to control speed.

This means there are two inputs each with two possible values. therefore there are 4 possible input states which are listed in table D.1

Table D.1: four possible input states to anti shoot through circuit

State	Direction line	PWM line
1	0	0
2	0	1
3	1	0
4	1	1

In order to confirm that the circuit performs the desired action it was simulated for each of the possible input cases to ensure that shoot through could not occur. the results of the simulation can be seen in figures D.1, D.2, D.3 and D.4.
In the simulations a red LED signifies a closed transistor and a grey LED signifies an open transistor these LEDs are included simply to make the outcomes of the circuit easily apparent. In each of these simulations it is confirmed that shoot through does not occur as the lights that indicate a closed MOSFET are never on on the same side of the H-bridge when controlled through the anti shoot through circuit.

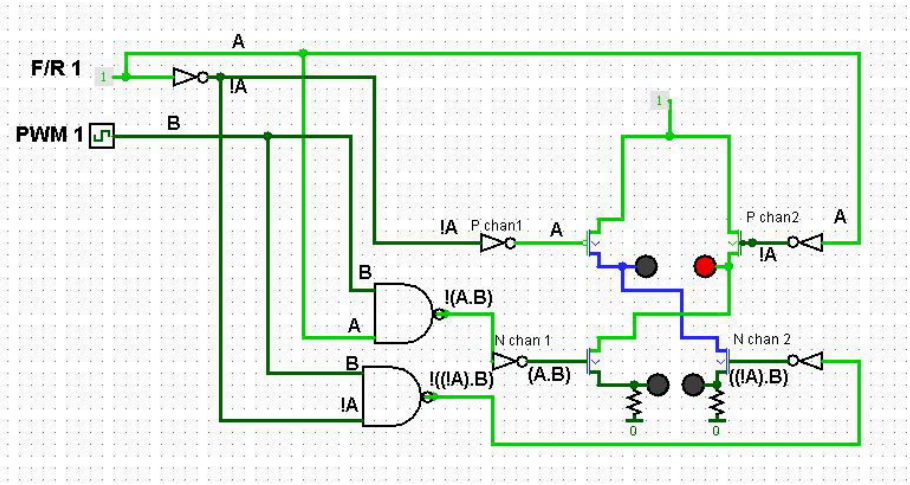


Figure D.1: Logisim simulation of designed circuit: F/R bit high, PWM low

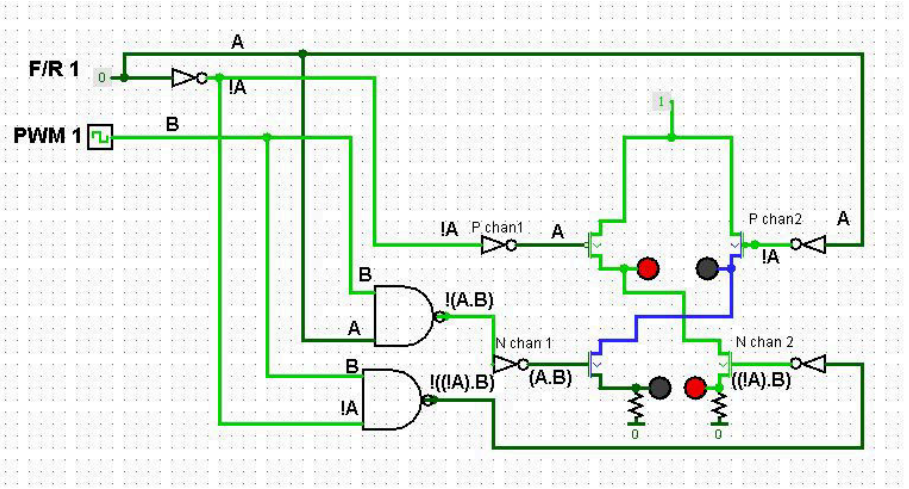


Figure D.2: Logisim simulation of designed circuit: F/R bit high, PWM high

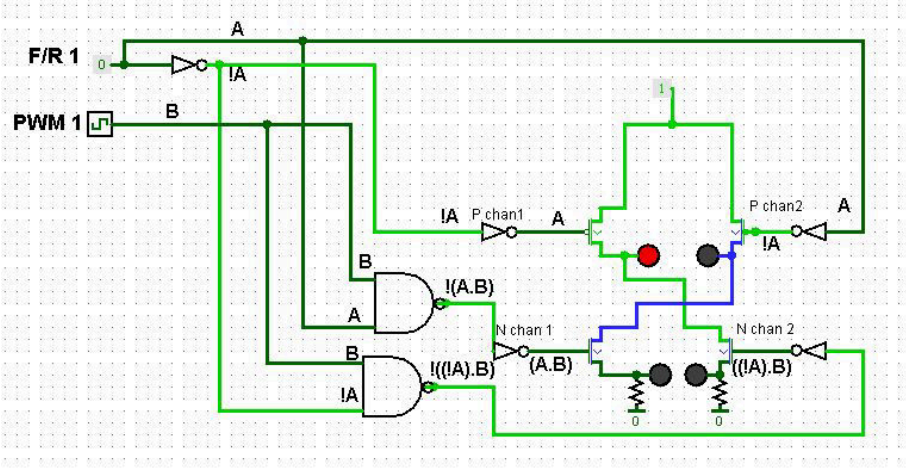


Figure D.3: Logisim simulation of designed circuit: F/R bit low, PWM low

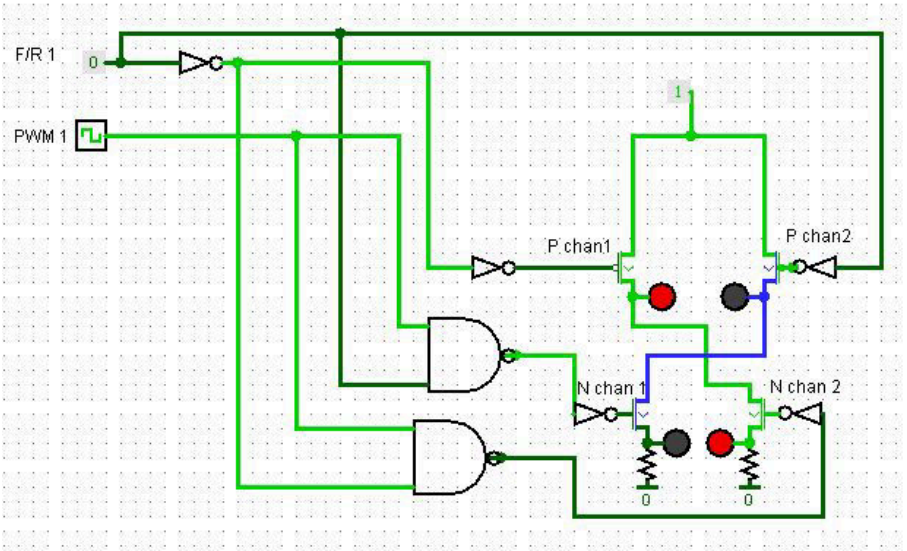


Figure D.4: Logisim simulation of designed circuit: F/R bit low, PWM high

E [Micro controller C code]

E.1 main.c file

```
/*
******************************************************************************
File:    main.c
Info:    Generated by Atollic TrueSTUDIO(R) 6.0.0 2017-09-28

The MIT License (MIT)
Copyright (c) 2009-2016 Atollic AB

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
******************************************************************************
*/

/* Includes */
#include "stm32f0xx.h"
#include <stdio.h>
#include "spi_imu.h"
#include <stdlib.h>
/* Private macro */
/* Private variables */
/* Private function prototypes */
```

```
/* Private functions */
void DELAY(uint32_t delay_in_us);
void init_PWM(void);
void init_Input_Cap(void);
void init_spi_ssm(void);
uint8_t read_from_address(uint16_t , uint32_t , uint32_t );
void write_to_address(uint16_t , uint8_t , uint32_t , uint32_t );
void set_PWM( int , int );
void Enable_drive(void);
void drive_left ( );
void drive_right ( );
uint32_t sel;
uint32_t deselect;
uint16_t address;
uint8_t value;
int8_t value0;
int8_t value1;
int16_t value_full;
uint16_t value_16;
int16_t thresholdup;
int16_t thresholddown;
int16_t time_old;
int16_t time_new;
uint input_cap [ 10 ];
uint8_t right_dir;
uint8_t left_dir;
uint8_t i;
uint8_t direction;
int8_t left_speed;
int8_t right_speed;
int8_t left_0;
int8_t right_0;
int8_t left_1;
int8_t right_1;
int8_t left_2;
int8_t right_2;
int8_t left;
int8_t right;
/**
**===========================================================================
**
** Abstract: main program
**
**===========================================================================
*/
```

```
void main (void)
{
  DELAY(3000);
  init_spi_ssm();
  init_PWM();
  Enable_drive();
  init_Input_Cap();
  DELAY(5000);
  value = 0x01;
  sel = GPIO_BSRR_BR_15;
  deselect=GPIO_BSRR_BS_15;
  address=0x6B;
  thresholdup = 5000;
  thresholddown = -5000;
  write_to_address( address, value , sel, deselect);
  direction = 0;
  GPIOA->ODR |= 0b10000;
  right_0=0;
  right_1=0;
  right_2=0;
  left_0=0;
  left_1=0;
  left_2=0;

  for(;;){
    drive_left ( );
    drive_right ( );
    set_PWM( left_speed, right_speed );
    sel = GPIO_BSRR_BR_15;
    deselect=GPIO_BSRR_BS_15;
    GPIOB->MODER |= GPIO_MODER_MODER0_0; //set PB0 to output
    GPIOB->MODER |= GPIO_MODER_MODER1_0; //set PB1 to output
    GPIOB->MODER |= GPIO_MODER_MODER2_0; //set PB2 to output

    address=0x40;
    value0 = read_from_address(address, sel, deselect);
    address=0x3F;
    value1 = read_from_address(address, sel, deselect);
    value_full = value1 << 8 | value0 & 1<<16;
    address=0x40;
    value_16 = read_from_address(address, sel, deselect);
    if (value_full > thresholdup) GPIOB->ODR = 0b1;
    else if (value_full < thresholddown) GPIOB->ODR = 0b10;
    else GPIOB->ODR = 0;

  }
```

```
}

void DELAY(uint32_t delay_in_us) {
  /* Hangs for specified number of microseconds. */
  volatile uint32_t counter = 0;
  delay_in_us *= 3;
  for(; counter < delay_in_us; counter++) {
    __asm("nop");
    __asm("nop");
  }
}

void init_PWM(void)
{
  RCC->AHBENR |= RCC_AHBENR_GPIOBEN;
  //RCC->AHBENR |= RCC_AHBENR_GPIOAEN;
  RCC->APB1ENR |= RCC_APB1ENR_TIM2EN;

  GPIOB->MODER |= GPIO_MODER_MODER10_1; // PB10 = AF
  GPIOB->MODER |= GPIO_MODER_MODER11_1; // PB11 = AF
  //GPIOA->MODER |= GPIO_MODER_MODER0_1; // PA0
  //GPIOA->MODER |= GPIO_MODER_MODER1_1; //PA1

  GPIOB->AFR[1] |= (2<<(4*(10-8))); // PB10_AF = AF2 (ie: map to TIM2_CH3)
  GPIOB->AFR[1] |= (2 << (4*(11 - 8))); // PB11_AF = AF2 (ie: map to TIM2_CH4)
  //GPIOA->AFR[0] |= (2); // PB10_AF = AF2 (ie: map to TIM2_CH3)
  //GPIOA->AFR[0] |= (0b0010<< 4); // PB11_AF = AF2 (ie: map to TIM2_CH4)


  TIM2->ARR = 8000; // f = 1 KHz
  // specify PWM mode: OCxM bits in CCMRx. We want mode 1
  TIM2->CCMR2 |= (TIM_CCMR2_OC3M_2 | TIM_CCMR2_OC3M_1); // PWM Mode 1
  TIM2->CCMR2 |= (TIM_CCMR2_OC4M_2 | TIM_CCMR2_OC4M_1); // PWM Mode 1
  //TIM2->CCMR1 |= (TIM_CCMR1_OC1M_2 | TIM_CCMR1_OC1M_1); // PWM Mode 1
  //TIM2->CCMR1 |= (TIM_CCMR1_OC2M_2 | TIM_CCMR1_OC2M_1); // PWM Mode 1
  // set PWM percentages
}

void set_PWM( int PB_10, int PB_11 )
{
   if (PB_10<100 && PB_11<100)
   {
      //TIM2->CCR1 = (x_1) * 80; // Percentage value given here is inverted
          when its actually used
      //TIM2->CCR2 = (y_1) * 80;
      TIM2->CCR3 = (PB_10) * 80; // Percentage value given here is inverted
          when its actually used
```

```
      TIM2->CCR4 = (PB_11) * 80;
      // enable the OC channels
      //TIM2->CCER |= TIM_CCER_CC1E;
      //TIM2->CCER |= TIM_CCER_CC2E;
      TIM2->CCER |= TIM_CCER_CC3E;
      TIM2->CCER |= TIM_CCER_CC4E;
      TIM2->CR1 |= TIM_CR1_CEN;// counter enable
   }
}

void init_Input_Cap(void)
{
   RCC -> AHBENR |= RCC_AHBENR_GPIOAEN; // enable clock PORT A
   RCC -> APB2ENR |= RCC_APB2ENR_TIM1EN; // enable clock for TIM14
   GPIOA -> MODER |= GPIO_MODER_MODER8_1; // set PA8 to AF mode
   GPIOA ->PUPDR |= GPIO_PUPDR_PUPDR8_1;
   GPIOA -> AFR[1] |= 0b0010; // set bit 2 of AFR8[3:0]
   TIM1 -> CCMR1 |= TIM_CCMR1_CC1S_0; // configure TIM1 for IC
   TIM1 -> CCMR1 |= (TIM_CCMR1_IC1F_0 | TIM_CCMR1_IC1F_1); // setup input
       filter
   TIM1 -> CCMR1 &= ~TIM_CCMR1_IC1PSC; // set PSC = 0
   TIM1 -> CCER &= ~(TIM_CCER_CC1NP | TIM_CCER_CC1P); // event = rising edge
   TIM1 -> CCER |= TIM_CCER_CC2P; // event = rising edge
   TIM1 -> PSC = 7; // Assuming fCLK = 48MHz
   TIM1 -> ARR = 59999; // TIM14 overflows every 0.01s
   TIM1 -> DIER |= TIM_DIER_CC1IE; // enable interrupt for input capture
   NVIC_EnableIRQ(TIM1_CC_IRQn); // unmask the interrupt in the NVIC
   TIM1 -> CCER |= TIM_CCER_CC1E; // input capture enabled
   TIM1 -> CR1 |= TIM_CR1_CEN;
}

void TIM1_CC_IRQHandler(void)
{

   time_old = time_new;
   time_new = TIM1->CCR1;
   if ((time_new-time_old)<15000)                    //filter out the
       receiver being off
   {
      if ((time_new-time_old)<2000)                 //check if pause pulse or
          signal pulse
      {
      input_cap [ i ]= (time_new-time_old);          //signal pulse
      i ++;
      }
      else if ((time_new-time_old)>2000)             //pause pulse
      {
```

```
input_cap [ i ] = 0;
i = 0;
input_cap [ i ]= (time_new-time_old);
}


right_2=right_1;
right_1=right_0;
right_0=(input_cap[2]/4)-379;                          //signal for
    right wheel (set from -100 to 100)
if (right_0>(right_1-10)||right_0<(right_1+10))
{
  if(right_1>(right_2-10)||right_1<(right_2+10))
  {
    right=right_0;
    if (right>20)                                  //sort positive and
        negative signals
    {
      right_speed=right;
      right_dir=0;
    }
    else if (right<-20)
    {
      right_speed=right*(-1);
      right_dir=1;
    }
    else if (right>-20&&right<20)                   //dead band to stop
        in middle
    {
      right_speed=0;
    }
  }
}



left_2=left_1;
left_1=left_0;
left_0=(input_cap[1]/4)-379;                           //signal for left
    wheel (set from -100 to 100)
if (left_0>(left_1-10)||left_0<(left_1+10))
{
  if(left_1>(left_2-10)||left_1<(left_2+10))
  {
    left=left_0;
```

```
          if (left>20)                                    //sort positive and
              negative signals
          {
             left_speed=left;
             left_dir=0;
          }
          else if (left<-20)
          {
             left_speed=left*(-1);
             left_dir=1;
          }
          else if (left>-20&&left<20)                     //dead band to stop
              in middle
          {
             left_speed=0;
          }
        }
      }



      if (right <-990 )
      {
        GPIOA->ODR &= ~0b11111;
      }

      else if (right >-990 )
      {
        GPIOA->ODR |= 0b10000;
      }
    }
}

void Enable_drive(void)
{
   RCC->AHBENR |= RCC_AHBENR_GPIOAEN;
   GPIOA->MODER |= GPIO_MODER_MODER0_0; // PA0
   GPIOA->MODER |= GPIO_MODER_MODER1_0; //PA1
   GPIOA->MODER |= GPIO_MODER_MODER2_0; // PA0
   GPIOA->MODER |= GPIO_MODER_MODER3_0; //PA1
   GPIOA->MODER |= GPIO_MODER_MODER4_0; // PA0


}

void drive_right (void)
{
```

```
   if (right_dir == 1)
      {
        GPIOA->ODR &= ~0b11;
        GPIOA->ODR |= 0b10;
      }

      else if (right_dir == 0)
      {
        GPIOA->ODR &= ~0b11;
        GPIOA->ODR |= 0b01;
      }
}

void drive_left (void)
{
   if (left_dir == 1)
      {
        GPIOA->ODR &= ~0b1100;
        GPIOA->ODR |= 0b1000;
      }

      else if (left_dir == 0)
      {
        GPIOA->ODR &= ~0b1100;
        GPIOA->ODR |= 0b0100;
      }
}
```

E.2 spi_imu.c file

```
#include "spi_imu.h"
#include "stm32f0xx.h"

static void DELAY(uint32_t delay_in_us);

/**
  * @brief Initializes the SPI/IMUBP communication.
  * @param None
  * @retval None
  */
void init_spi_ssm(void) {
   // clock enable to GPIO
   RCC->AHBENR |= RCC_AHBENR_GPIOAEN;   //enable clock for PA bus
   RCC->AHBENR |= RCC_AHBENR_GPIOBEN;   //enable clock for PB bus

   // mode for cs pins
   GPIOA->MODER |= GPIO_MODER_MODER10_0; // PA10:CS_bmx055_mag output
   GPIOA->MODER |= GPIO_MODER_MODER12_0; // PA12:CS_bmx055_gyr output
   GPIOA->MODER |= GPIO_MODER_MODER15_0; // PA15:CS_bmx055_acc output
   // pull cs high
   GPIOA->BSRR |= (GPIO_BSRR_BS_10|GPIO_BSRR_BS_12|GPIO_BSRR_BS_15);

   // mode for spi pins
   // no need to map pins to the perpiheral - they are AF0
   GPIOB->MODER |= GPIO_MODER_MODER3_1;   // set pin SCK (PB3) to Alternate
       Function
   GPIOB->MODER |= GPIO_MODER_MODER4_1;   // set pin MISO (PB4) to Alternate
       Function
   GPIOB->MODER |= GPIO_MODER_MODER5_1;   // set pin MOSI (PB5) to Alternate
       Function

   // clock enable to SPI1
   RCC->APB2ENR |= RCC_APB2ENR_SPI1EN;

   // spi configuration
   SPI1->CR1 |= SPI_CR1_BIDIOE;     // enable output
   SPI1->CR1 |= (SPI_CR1_BR_0 | SPI_CR1_BR_1|SPI_CR1_BR_2); // set baud to
       fpclk 0.8 MHz for 48 MHz clock
   SPI1->CR1 |= SPI_CR1_MSTR;          // set to master mode
   SPI1->CR2 |= SPI_CR2_FRXTH;      // set RX threshold to be 8 bits

   SPI1->CR1 |= SPI_CR1_CPOL;
   SPI1->CR1 |= SPI_CR1_CPHA;
```

```
    SPI1->CR1 |= SPI_CR1_SSM;           // software NNS management
    SPI1->CR1 |= SPI_CR1_SSI;         //pull it high

    SPI1->CR2 |= (SPI_CR2_DS_0 | SPI_CR2_DS_1 | SPI_CR2_DS_2); // set to 8bit
        mode
    SPI1->CR1 |= SPI_CR1_SPE;         // enable the SPI peripheral
}


/**
  * @brief Writes a block on the BMX055/BMP280
  * @param address: register address to access
  * @param data: data to write on the register
  * @param select: chip select
  * @param deselect: chip deselect
  * @retval None
  */
void write_to_address(uint16_t address, uint8_t data, uint32_t select,
    uint32_t deselect) {
  uint8_t dummy;                  // a variable to hold junk data from DR

  // trasmit the address to write to
  GPIOA->BSRR |= select;            // pull CS low
  DELAY(1);
  *((uint8_t*)(&SPI1->DR)) = ((0x7F)&address);;
  while ((SPI1->SR & SPI_SR_RXNE) == 0);  // hang while RX is empty
  dummy = SPI1->DR;

  // trasmit the data to write in the address
  *((uint8_t*)(&SPI1->DR)) = data;     // address MSB
  while ((SPI1->SR & SPI_SR_RXNE) == 0);   // hang while RX is empty
  dummy = SPI1->DR;

  GPIOA->BSRR |= deselect;            // pull CS high
  DELAY(5000);
}


/**
  * @brief Reads a block on the BMX055/BMP280
  * @param address: register address to access
  * @param data: data to read from the register
  * @param enble: chip select BSRR u32
  * @param deselect: chip deselect BSRR u32
  * @retval data
  */
```

```
uint8_t read_from_address(uint16_t address, uint32_t select, uint32_t
    deselect) {
  uint16_t dummy;                    // a variable to hold junk data from DR

  // trasmit the address to read from
  GPIOA->BSRR |= select;             // pull CS low
  DELAY(1);
  *((uint8_t*)(&SPI1->DR)) = ((0x80)|address);
  while ((SPI1->SR & SPI_SR_RXNE) == 0);   // hang while RX is empty
  dummy = SPI1->DR;

   // receive the data form the address
  *((uint8_t*)(&SPI1->DR)) = 0x42;     // clock out some junk data
  while ((SPI1->SR & SPI_SR_RXNE) == 0);
  dummy = SPI1->DR;


  GPIOA->BSRR |= deselect;             // pull CS high
  DELAY(5000);
  return dummy;
}


static void DELAY(uint32_t delay_in_us) {
  /* Hangs for specified number of microseconds. */
  volatile uint32_t counter = 0;
  delay_in_us *= 3;
  for(; counter < delay_in_us; counter++) {
    __asm("nop");
    __asm("nop");
  }
}
```

E.3 BMP280.h file

```
#ifndef BMP280_H
#define BMP280_H

/**
  SPI 4 wire interface
  Maximum frequency (VDDIO>1.75):  10 MHz
  Automatic selection (CPOL,CPHA):  (0,0) & (1,1)
  CSB status:               idle 1 & active 0
**/


//------------------------------REGISTERS------------------------------//
// Barometer registers
#define BMP280_PRS_CALIB00 0x88 // should return 0x58
#define BMP280_PRS_CALIB01 0x89
#define BMP280_PRS_CALIB02 0x8A
#define BMP280_PRS_CALIB03 0x8B
#define BMP280_PRS_CALIB04 0x8C
#define BMP280_PRS_CALIB05 0x8D
#define BMP280_PRS_CALIB06 0x8E
#define BMP280_PRS_CALIB07 0x8F
#define BMP280_PRS_CALIB08 0x90
#define BMP280_PRS_CALIB09 0x91
#define BMP280_PRS_CALIB10 0x92
#define BMP280_PRS_CALIB11 0x93
#define BMP280_PRS_CALIB12 0x94
#define BMP280_PRS_CALIB13 0x95
#define BMP280_PRS_CALIB14 0x96
#define BMP280_PRS_CALIB15 0x97
#define BMP280_PRS_CALIB16 0x98
#define BMP280_PRS_CALIB17 0x99
#define BMP280_PRS_CALIB18 0x9A
#define BMP280_PRS_CALIB19 0x9B
#define BMP280_PRS_CALIB20 0x9C
#define BMP280_PRS_CALIB21 0x9D
#define BMP280_PRS_CALIB22 0x9E
#define BMP280_PRS_CALIB23 0x9F
#define BMP280_PRS_CALIB24 0xA0
#define BMP280_PRS_CALIB25 0xA1

#define BMP280_PRS_CHIP_ID 0xD0
#define BMP280_PRS_RESET 0xE0
#define BMP280_PRS_STATUS 0xF3
#define BMP280_PRS_CTRL_MEAS 0xF4
```

```
#define BMP280_PRS_CONFIG 0xF5
#define BMP280_PRS_PRESS_MSB 0xF7
#define BMP280_PRS_PRESS_LSB 0xF8
#define BMP280_PRS_PRESS_XLSB 0xF9
#define BMP280_PRS_TEMP_MSB 0xFA
#define BMP280_PRS_TEMP_LSB 0xFB
#define BMP280_PRS_TEMP_XLSB 0xFC

//------------------------------DEVICE_ID------------------------------//

// eight-bit device ID are PRS = 0x58
#define BMX055_PRS_ADDRESS  0x58  // Address of BMP280


//---------------------BAROMETER REGISTER PIN_OUTS---------------------//
// BMP280_PRS_CHIP_ID  0xD0
#define PRS_CHIP_ID  0x58

// BMP280_PRS_RESET  0xE0
#define PRS_RESET_RESET  0xB6

// BMP280_PRS_STATUS  0xF3
#define PRS_STATUS_MEASURING 0x04
#define PRS_STATUS_IM_UPDATE 0x01

// BMP280_PRS_CTRL_MEAS  0xF4
#define PRS_CTRL_OSRS_T  0xE0
#define PRS_CTRL_OSRS_P  0x1C
#define PRS_CTRL_MODE  0x03

#define PRS_CTRL_OSRS_T_SKIP 0x00
#define PRS_CTRL_OSRS_T_1 0x20
#define PRS_CTRL_OSRS_T_2 0x40
#define PRS_CTRL_OSRS_T_4 0x60
#define PRS_CTRL_OSRS_T_8 0x80
#define PRS_CTRL_OSRS_T_16 0xA0

#define PRS_CTRL_OSRS_P_SKIP 0x00
#define PRS_CTRL_OSRS_P_1 0x04
#define PRS_CTRL_OSRS_P_2 0x08
#define PRS_CTRL_OSRS_P_4 0x0C
#define PRS_CTRL_OSRS_P_8 0x10
#define PRS_CTRL_OSRS_P_16 0x14

// BMP280_PRS_CONFIG  0xF5
#define PRS_CONFIG_T_SB  0xE0
#define PRS_CONFIG_FILTER 0x1C
```

```
#define PRS_CONFIG_SPI3W_EN 0x01

#define PRS_CONFIG_T_SB_500us 0x00
#define PRS_CONFIG_T_SB_62_5ms 0x20
#define PRS_CONFIG_T_SB_125ms 0x40
#define PRS_CONFIG_T_SB_250ms 0x60
#define PRS_CONFIG_T_SB_500ms 0x80
#define PRS_CONFIG_T_SB_1s 0xA0
#define PRS_CONFIG_T_SB_2s 0xC0
#define PRS_CONFIG_T_SB_4s 0xE0

#define PRS_CONFIG_FILTER_OFF 0x00
#define PRS_CONFIG_FILTER_2 0x04
#define PRS_CONFIG_FILTER_4 0x08
#define PRS_CONFIG_FILTER_8 0x10
#define PRS_CONFIG_FILTER_16 0x1C

// BMP280_PRS_PRESS_MSB  0xF7
#define PRS_PRESS_MSB_MSB 0xFF

// BMP280_PRS_PRESS_LSB  0xF8
#define PRS_PRESS_LSB_LSB 0xFF

// BMP280_PRS_PRESS_XLSB 0xF9
#define PRS_PRESS_XLSB_XLSB 0xF0

// BMP280_PRS_TEMP_MSB  0xFA
#define PRS_TEMP_MSB_MSB 0xFF

// BMP280_PRS_TEMP_LSB  0xFB
#define PRS_TEMP_LSB_LSB 0xFF

// BMP280_PRS_TEMP_XLSB  0xFC
#define PRS_TEMP_XLSB_XLSB 0xF0


#endif
```

E.4 BMX055.h file

```
#ifndef BMX055_H
#define BMX055_H


/**
  SPI 4 wire interface
  Maximum frequency (VDDIO>1.75):  10 MHz
  Automatic selection (CPOL,CPHA):  (0,0) & (1,1)
  CSB status:          idle 1 & active 0
**/

//------------------------------REGISTERS------------------------------//
// Accelerometer registers
#define BMX055_ACC_CHIP_ID  0x00 // should return 0xFA
//#define BMX055_ACC_Reserved  0x01
#define BMX055_ACC_D_X_LSB  0x02
#define BMX055_ACC_D_X_MSB  0x03
#define BMX055_ACC_D_Y_LSB  0x04
#define BMX055_ACC_D_Y_MSB  0x05
#define BMX055_ACC_D_Z_LSB  0x06
#define BMX055_ACC_D_Z_MSB  0x07
#define BMX055_ACC_D_TEMP  0x08
#define BMX055_ACC_INT_STATUS_0  0x09
#define BMX055_ACC_INT_STATUS_1  0x0A
#define BMX055_ACC_INT_STATUS_2  0x0B
#define BMX055_ACC_INT_STATUS_3  0x0C
//#define BMX055_ACC_Reserved  0x0D
#define BMX055_ACC_FIFO_STATUS  0x0E
#define BMX055_ACC_PMU_RANGE  0x0F
#define BMX055_ACC_PMU_BW  0x10
#define BMX055_ACC_PMU_LPW  0x11
#define BMX055_ACC_PMU_LOW_POWER 0x12
#define BMX055_ACC_D_HBW  0x13
#define BMX055_ACC_BGW_SOFTRESET 0x14
//#define BMX055_ACC_Reserved  0x15
#define BMX055_ACC_INT_EN_0  0x16
#define BMX055_ACC_INT_EN_1  0x17
#define BMX055_ACC_INT_EN_2  0x18
#define BMX055_ACC_INT_MAP_0  0x19
#define BMX055_ACC_INT_MAP_1  0x1A
#define BMX055_ACC_INT_MAP_2  0x1B
//#define BMX055_ACC_Reserved  0x1C
//#define BMX055_ACC_Reserved  0x1D
#define BMX055_ACC_INT_SRC  0x1E
```

```
//#define BMX055_ACC_Reserved  0x1F
#define BMX055_ACC_INT_OUT_CTRL  0x20
#define BMX055_ACC_INT_RST_LATCH 0x21
#define BMX055_ACC_INT_0  0x22
#define BMX055_ACC_INT_1  0x23
#define BMX055_ACC_INT_2  0x24
#define BMX055_ACC_INT_3  0x25
#define BMX055_ACC_INT_4  0x26
#define BMX055_ACC_INT_5  0x27
#define BMX055_ACC_INT_6  0x28
#define BMX055_ACC_INT_7  0x29
#define BMX055_ACC_INT_8  0x2A
#define BMX055_ACC_INT_9  0x2B
#define BMX055_ACC_INT_A  0x2C
#define BMX055_ACC_INT_B  0x2D
#define BMX055_ACC_INT_C  0x2E
#define BMX055_ACC_INT_D  0x2F
#define BMX055_ACC_FIFO_CONFIG_0 0x30
//#define BMX055_ACC_Reserved  0x31
#define BMX055_ACC_PMU_SELF_TEST 0x32
#define BMX055_ACC_TRIM_NVM_CTRL 0x33
#define BMX055_ACC_BGW_SPI3_WDT  0x34
//#define BMX055_ACC_Reserved  0x35
#define BMX055_ACC_OFC_CTRL  0x36
#define BMX055_ACC_OFC_SETTING  0x37
#define BMX055_ACC_OFC_OFFSET_X  0x38
#define BMX055_ACC_OFC_OFFSET_Y  0x39
#define BMX055_ACC_OFC_OFFSET_Z  0x3A
#define BMX055_ACC_TRIM_GP0  0x3B
#define BMX055_ACC_TRIM_GP1  0x3C
//#define BMX055_ACC_Reserved  0x3D
#define BMX055_ACC_FIFO_CONFIG_1 0x3E
#define BMX055_ACC_FIFO_DATA  0x3F

// BMX055 Gyroscope Registers
#define BMX055_GYRO_CHIP_ID  0x00 // should return 0x0F
//#define BMX055_GYRO_Reserved  0x01
#define BMX055_GYRO_RATE_X_LSB  0x02
#define BMX055_GYRO_RATE_X_MSB  0x03
#define BMX055_GYRO_RATE_Y_LSB  0x04
#define BMX055_GYRO_RATE_Y_MSB  0x05
#define BMX055_GYRO_RATE_Z_LSB  0x06
#define BMX055_GYRO_RATE_Z_MSB  0x07
//#define BMX055_GYRO_Reserved  0x08
#define BMX055_GYRO_INT_STATUS_0 0x09
#define BMX055_GYRO_INT_STATUS_1 0x0A
#define BMX055_GYRO_INT_STATUS_2 0x0B
```

```
#define BMX055_GYRO_INT_STATUS_3 0x0C
//#define BMX055_GYRO_Reserved  0x0D
#define BMX055_GYRO_FIFO_STATUS  0x0E
#define BMX055_GYRO_RANGE  0x0F
#define BMX055_GYRO_BW    0x10
#define BMX055_GYRO_LPM1  0x11
#define BMX055_GYRO_LPM2  0x12
#define BMX055_GYRO_RATE_HBW   0x13
#define BMX055_GYRO_BGW_SOFTRESET 0x14
#define BMX055_GYRO_INT_EN_0  0x15
#define BMX055_GYRO_INT_EN_1  0x16
#define BMX055_GYRO_INT_MAP_0  0x17
#define BMX055_GYRO_INT_MAP_1  0x18
#define BMX055_GYRO_INT_MAP_2  0x19
#define BMX055_GYRO_INT_SRC_1  0x1A
#define BMX055_GYRO_INT_SRC_2  0x1B
#define BMX055_GYRO_INT_SRC_3  0x1C
//#define BMX055_GYRO_Reserved  0x1D
#define BMX055_GYRO_FIFO_EN  0x1E
//#define BMX055_GYRO_Reserved  0x1F
//#define BMX055_GYRO_Reserved  0x20
#define BMX055_GYRO_INT_RST_LATCH 0x21
#define BMX055_GYRO_HIGH_TH_X  0x22
#define BMX055_GYRO_HIGH_DUR_X   0x23
#define BMX055_GYRO_HIGH_TH_Y  0x24
#define BMX055_GYRO_HIGH_DUR_Y  0x25
#define BMX055_GYRO_HIGH_TH_Z  0x26
#define BMX055_GYRO_HIGH_DUR_Z  0x27
//#define BMX055_GYRO_Reserved  0x28
//#define BMX055_GYRO_Reserved  0x29
//#define BMX055_GYRO_Reserved  0x2A
//#define BMX055_GYRO_Reserved  0x2B
//#define BMX055_GYRO_Reserved  0x2C
//#define BMX055_GYRO_Reserved  0x2D
//#define BMX055_GYRO_Reserved  0x2E
//#define BMX055_GYRO_Reserved  0x2F
//#define BMX055_GYRO_Reserved  0x30
#define BMX055_GYRO_SOC   0x31
#define BMX055_GYRO_A_FOC  0x32
#define BMX055_GYRO_TRIM_NVM_CTRL 0x33
#define BMX055_GYRO_BGW_SPI3_WDT 0x34
//#define BMX055_GYRO_Reserved  0x35
#define BMX055_GYRO_OFC1  0x36
#define BMX055_GYRO_OFC2  0x37
#define BMX055_GYRO_OFC3  0x38
#define BMX055_GYRO_OFC4  0x39
#define BMX055_GYRO_TRIM_GP0  0x3A
```

```
#define BMX055_GYRO_TRIM_GP1  0x3B
#define BMX055_GYRO_BIST  0x3C
#define BMX055_GYRO_FIFO_CONFIG_0 0x3D
#define BMX055_GYRO_FIFO_CONFIG_1 0x3E
#define BMX055_GYRO_FIFO_DATA  0x3F

// BMX055 magnetometer registers
#define BMX055_MAG_CHIP_ID  0x40 // should return 0x32
//define BMX055_MAG_Reserved  0x41
#define BMX055_MAG_XOUT_LSB  0x42
#define BMX055_MAG_XOUT_MSB  0x43
#define BMX055_MAG_YOUT_LSB  0x44
#define BMX055_MAG_YOUT_MSB  0x45
#define BMX055_MAG_ZOUT_LSB  0x46
#define BMX055_MAG_ZOUT_MSB  0x47
#define BMX055_MAG_ROUT_LSB  0x48
#define BMX055_MAG_ROUT_MSB  0x49
#define BMX055_MAG_INT_STATUS  0x4A
#define BMX055_MAG_PWR_CR1  0x4B
#define BMX055_MAG_PWR_CR2  0x4C
#define BMX055_MAG_INT_CR_1  0x4D
#define BMX055_MAG_INT_CR_2  0x4E
#define BMX055_MAG_INT_LOWTHS_CR 0x4F
#define BMX055_MAG_INT_HIGHTHS_CR 0x50
#define BMX055_MAG_REP_XY  0x51
#define BMX055_MAG_REP_Z  0x52

//------------------------------DEVICE_ID------------------------------//

// eight-bit device ID are ACC = 0xFA, GYRO = 0x0F, MAG = 0x32
#define BMX055_ACC_ADDRESS 0xFA  // Address of BMX055 accelerometer
#define BMX055_GYRO_ADDRESS 0x0F  // Address of BMX055 gyroscope
#define BMX055_MAG_ADDRESS 0x32  // Address of BMX055 magnetometer

//-------------------ACCELEROMETER REGISTER PIN_OUTS-------------------//

// BMX055_ACC_CHIP_ID       0x00
#define ACC_CHIP_ID_CHIP_ID    0xFF

// BMX055_ACC_Reserved      0x01

// BMX055_ACC_D_X_LSB       0x02
#define ACC_D_X_LSB_X_LSB      0xF0
#define ACC_D_X_LSB_NEW_DATA   0x01

// BMX055_ACC_D_X_MSB     0x03
#define ACC_D_X_MSB_X_MSB   0xFF
```

```
// BMX055_ACC_D_Y_LSB       0x04
#define ACC_D_Y_LSB_Y_LSB   0xF0
#define ACC_D_Y_LSB_NEW_DATA   0x01

// BMX055_ACC_D_Y_MSB       0x05
#define ACC_D_Y_MSB_Y_MSB   0xFF

// BMX055_ACC_D_Z_LSB       0x06
#define ACC_D_Z_LSB_Z_LSB   0xF0
#define ACC_D_Z_LSB_NEW_DATA   0x01

// BMX055_ACC_D_Z_MSB       0x07
#define ACC_D_Z_MSB_Z_MSB   0xFF

// BMX055_ACC_D_TEMP      0x08
#define ACC_D_TEMP_TEMP        0xFF

// BMX055_ACC_INT_STATUS_0     0x09
#define ACC_INT_STATUS_0_FLAT   0x80
#define ACC_INT_STATUS_0_ORIENT   0x40
#define ACC_INT_STATUS_0_S_TAP   0x20
#define ACC_INT_STATUS_0_D_TAP   0x10
#define ACC_INT_STATUS_0_SLO_NOT_MOT 0x08
#define ACC_INT_STATUS_0_SLOPE   0x04
#define ACC_INT_STATUS_0_HIGH   0x02
#define ACC_INT_STATUS_0_INT   0x01

// BMX055_ACC_INT_STATUS_1     0x0A
#define ACC_INT_STATUS_1_DATA   0x80
#define ACC_INT_STATUS_1_FIFO_WM  0x40
#define ACC_INT_STATUS_1_FIFO_FULL  0x20

// BMX055_ACC_INT_STATUS_2     0x0B
#define ACC_INT_STATUS_0_TAP_SIGN  0x80
#define ACC_INT_STATUS_0_TAP_F_Z  0x40
#define ACC_INT_STATUS_0_TAP_F_Y  0x20
#define ACC_INT_STATUS_0_TAP_F_X  0x10
#define ACC_INT_STATUS_0_SLOPE_SIGN  0x08
#define ACC_INT_STATUS_0_SLOPE_F_Z  0x04
#define ACC_INT_STATUS_0_SLOPE_F_Y  0x02
#define ACC_INT_STATUS_0_SLOPE_F_X  0x01

// BMX055_ACC_INT_STATUS_3     0x0C
#define ACC_INT_STATUS_3_FLAT   0x80
#define ACC_INT_STATUS_3_ORIENT_Z  0x40
#define ACC_INT_STATUS_0_ORIENT_YX  0x30
```

```
#define ACC_INT_STATUS_0_HIGH_SIGN  0x08
#define ACC_INT_STATUS_0_HIGH_F_Z  0x04
#define ACC_INT_STATUS_0_HIGH_F_Y  0x02
#define ACC_INT_STATUS_0_HIGH_F_X  0x01

// BMX055_ACC_Reserved         0x0D

// BMX055_ACC_FIFO_STATUS      0x0E
#define ACC_FIFO_STATUS_OVERRUN   0x80
#define ACC_FIFO_STATUS_FRAME_CNT  0xEF

// BMX055_ACC_PMU_RANGE        0x0F
#define ACC_PMU_RANGE_02       0x03
#define ACC_PMU_RANGE_04       0x05
#define ACC_PMU_RANGE_08       0x08
#define ACC_PMU_RANGE_16       0x0C

// BMX055_ACC_PMU_BW        0x10
#define ACC_PMU_BW_7_81        0x08
#define ACC_PMU_BW_15_63       0x09
#define ACC_PMU_BW_31_14       0x0A
#define ACC_PMU_BW_62_5        0x0B
#define ACC_PMU_BW_125         0x0C
#define ACC_PMU_BW_250         0x0D
#define ACC_PMU_BW_500         0x0E
#define ACC_PMU_BW_1000        0x0F

#define ACC_PMU_ODR_15_625     0x08
#define ACC_PMU_ODR_31_25      0x09
#define ACC_PMU_ODR_62_5       0x0A
#define ACC_PMU_ODR_125        0x0B
#define ACC_PMU_ODR_250        0x0C
#define ACC_PMU_ODR_500        0x0D
#define ACC_PMU_ODR_1000       0x0E
#define ACC_PMU_ODR_2000       0x0F

// BMX055_ACC_PMU_LPW          0x11
#define ACC_PWR_NORMAL         0x00
#define ACC_PWR_DEEP_SUSPEND   0x20
#define ACC_PWR_LOW_POWER      0x40
#define ACC_PWR_SUSPEND        0x80
#define ACC_SLEEP_DUR_RESET      0b11100001
#define ACC_SLEEP_DUR_500us      0b00000000
#define ACC_SLEEP_DUR_500us_1   0b00001010
#define ACC_SLEEP_DUR_1ms      0b00001100
#define ACC_SLEEP_DUR_2ms      0b00001110
#define ACC_SLEEP_DUR_4ms      0b00010000
```

```
#define ACC_SLEEP_DUR_6ms       0b00010010
#define ACC_SLEEP_DUR_10ms      0b00010100
#define ACC_SLEEP_DUR_25ms      0b00010110
#define ACC_SLEEP_DUR_50ms      0b00011000
#define ACC_SLEEP_DUR_100ms       0b00011010
#define ACC_SLEEP_DUR_500ms       0b00011100
#define ACC_SLEEP_DUR_1s        0b00011110

// BMX055_ACC_PMU_LOW_POWER       0x12
#define ACC_PMU_LOW_POWER_RESET    0b10011111
#define ACC_PMU_LOW_POWER_LOWPOWER  0x40
#define ACC_PMU_LOW_POWER_SLEEPTIMER 0x20

// BMX055_ACC_D_HBW            0x13
#define ACC_D_HBW_DATA_RESET    0x3F
#define ACC_D_HBW_DATA_HIGH_BW    0x80
#define ACC_D_HBW_SHADOW_DIS    0x40

// BMX055_ACC_BGW_SOFTRESET       0x14
#define ACC_BWG_SOFTRESET_RESET    0x00
#define ACC_BWG_SOFTRESET       0xB6

// BMX055_ACC_Reserved         0x15

// BMX055_ACC_INT_EN_0         0x16
#define ACC_INT_EN_0_FLAT       0x80
#define ACC_INT_EN_0_ORIENT       0x40
#define ACC_INT_EN_0_S_TAP      0x20
#define ACC_INT_EN_0_D_TAP      0x10
#define ACC_INT_EN_0_RESET      0x08
#define ACC_INT_EN_0_SLOPE_Z    0x04
#define ACC_INT_EN_0_SLOPE_Y    0x02
#define ACC_INT_EN_0_SLOPE_X    0x01

// BMX055_ACC_INT_EN_1         0x17
#define ACC_INT_EN_1_RESET      0x80
#define ACC_INT_EN_1_FWM        0x40
#define ACC_INT_EN_1_FFULL      0x20
#define ACC_INT_EN_1_DATA       0x10
#define ACC_INT_EN_1_LOW        0x08
#define ACC_INT_EN_1_HIGH_Z       0x04
#define ACC_INT_EN_1_HIGH_Y       0x02
#define ACC_INT_EN_1_HIGH_X       0x01

// BMX055_ACC_INT_EN_2         0x18
#define ACC_INT_EN_2_RESET      0xF0
#define ACC_INT_EN_2_SLO_NO_MOT_SEL   0x08
```

```
#define ACC_INT_EN_2_SLO_NO_MOT_Z  0x04
#define ACC_INT_EN_2_SLO_NO_MOT_Y  0x02
#define ACC_INT_EN_2_SLO_NO_MOT_X  0x01

// BMX055_ACC_INT_MAP_0        0x19
#define ACC_INT_MAP_0_FLAT_1   0x80
#define ACC_INT_MAP_0_ORIENT_1   0x40
#define ACC_INT_MAP_0_S_TAP_1   0x20
#define ACC_INT_MAP_0_D_TAP_1   0x10
#define ACC_INT_MAP_0_SLO_NO_MOT_1  0x08
#define ACC_INT_MAP_0_SLOPE_1   0x04
#define ACC_INT_MAP_0_HIGH_1   0x02
#define ACC_INT_MAP_0_LOW_1      0x01

// BMX055_ACC_INT_MAP_1        0x1A
#define ACC_INT_MAP_1_RESET      0x18
#define ACC_INT_MAP_1_DATA_2   0x80
#define ACC_INT_MAP_1_FWM_2      0x40
#define ACC_INT_MAP_1_FFULL_2   0x20
#define ACC_INT_MAP_1_FFULL_1   0x04
#define ACC_INT_MAP_1_FWM_1      0x02
#define ACC_INT_MAP_1_DATA_1   0x01

// BMX055_ACC_INT_MAP_2        0x1B
#define ACC_INT_MAP_0_FLAT_2   0x80
#define ACC_INT_MAP_0_ORIENT_2   0x40
#define ACC_INT_MAP_0_S_TAP_2   0x20
#define ACC_INT_MAP_0_D_TAP_2   0x10
#define ACC_INT_MAP_0_SLO_NO_MOT_2  0x08
#define ACC_INT_MAP_0_SLOPE_2   0x04
#define ACC_INT_MAP_0_HIGH_2   0x02
#define ACC_INT_MAP_0_LOW_2      0x01

// BMX055_ACC_Reserved         0x1C
// BMX055_ACC_Reserved         0x1D

// BMX055_ACC_INT_SRC          0x1E
#define ACC_INT_SRC_DATA       0x20
#define ACC_INT_SRC_TAP        0x10
#define ACC_INT_SRC_SLO_NO_MOT   0x08
#define ACC_INT_SRC_SLOPE      0x04
#define ACC_INT_SRC_HIGH       0x02
#define ACC_INT_SRC_LOW        0x01

// BMX055_ACC_Reserved         0x1F

// BMX055_ACC_INT_OUT_CTRL     0x20
```

```
#define ACC_INT_OUT_CTRL_RESET   0xF0
#define ACC_INT_OUT_CTRL_2_OD    0x08
#define ACC_INT_OUT_CTRL_2_LVL   0x04
#define ACC_INT_OUT_CTRL_1_OD    0x02
#define ACC_INT_OUT_CTRL_1_LVL   0x01

// BMX055_ACC_INT_RST_LATCH      0x21
#define ACC_INT_RST_LATCH_RESET   0x80
#define ACC_INT_RST_LATCH_NON   0x00
#define ACC_INT_RST_LATCH_250ms   0x01
#define ACC_INT_RST_LATCH_500ms   0x02
#define ACC_INT_RST_LATCH_1s   0x03
#define ACC_INT_RST_LATCH_2s   0x04
#define ACC_INT_RST_LATCH_4s   0x05
#define ACC_INT_RST_LATCH_8s   0x06
#define ACC_INT_RST_LATCH_LATCHED  0x07
#define ACC_INT_RST_LATCH_NON_1   0x08
#define ACC_INT_RST_LATCH_250us   0x09
#define ACC_INT_RST_LATCH_500us   0x0A
#define ACC_INT_RST_LATCH_1ms   0x0B
#define ACC_INT_RST_LATCH_12_5ms  0x0C
#define ACC_INT_RST_LATCH_25ms   0x0D
#define ACC_INT_RST_LATCH_50ms   0x0E
#define ACC_INT_RST_LATCH_LATCHED1  0x0F


// BMX055_ACC_INT_0           0x22
#define ACC_INT_0_LOW_DUR      0xFF

// BMX055_ACC_INT_1           0x23
#define ACC_INT_1_LOW_TH       0xFF

// BMX055_ACC_INT_2           0x24
#define ACC_INT_2_HIGH_HY      0xC0
#define ACC_INT_2_LOW_MODE     0x04
#define ACC_INT_2_LOW_HY       0x03

// BMX055_ACC_INT_3           0x25
#define ACC_INT_3_HIGH_DUR     0xFF

// BMX055_ACC_INT_4           0x26
#define ACC_INT_4_HIGH_TH      0xFF

// BMX055_ACC_INT_5           0x27
#define ACC_INT_5_SLO_NO_MOT_DUR  0xFC
#define ACC_INT_5_SLOPE_DUR       0x03
```

```
// BMX055_ACC_INT_6            0x28
#define ACC_INT_6_SLOPE_TH     0xFF

// BMX055_ACC_INT_7            0x29
#define ACC_INT_7_SLO_NO_MOT_TH   0xFF

// BMX055_ACC_INT_8            0x2A
#define ACC_INT_8_TAP_QUIET      0x80
#define ACC_INT_8_TAP_SHOCK      0x40
#define ACC_INT_8_TAP_DUR      0x07

// BMX055_ACC_INT_9            0x2B
#define ACC_INT_9_TAP_SAMP     0xC0
#define ACC_INT_9_TAP_TH       0x1F

// BMX055_ACC_INT_A            0x2C
#define ACC_INT_A_ORIENT_HYST   0x70
#define ACC_INT_A_ORIENT_BLOCKING  0x0C
#define ACC_INT_A_ORIENT_MODE   0x03

// BMX055_ACC_INT_B            0x2D
#define ACC_INT_B_ORIENT_UD_EN   0x40
#define ACC_INT_B_ORIENT_THETA   0x3F

// BMX055_ACC_INT_C            0x2E
#define ACC_INT_C_FLAT_THETA   0x3F

// BMX055_ACC_INT_D            0x2F
#define ACC_INT_D_FLAT_HOLD_TIME  0x30
#define ACC_INT_D_FLAT_HY      0x07

// BMX055_ACC_FIFO_CONFIG_0       0x30
#define ACC_FIFO_CONFIG_WM_LEVEL  0x3F

// BMX055_ACC_Reserved         0x31

// BMX055_ACC_PMU_SELF_TEST       0x32
#define ACC_PMU_SELF_TEST_AMP   0x10
#define ACC_PMU_SELF_TEST_SIGN   0x04
#define ACC_PMU_SELF_TEST_AXIS   0x03

// BMX055_ACC_TRIM_NVM_CTRL       0x33
#define ACC_TRIM_NVM_CTRL_REMAIN  0xF0
#define ACC_TRIM_NVM_CTRL_LOAD   0x08
#define ACC_TRIM_NVM_CTRL_RDY   0x04
#define ACC_TRIM_NVM_CTRL_PROG_TRIG  0x02
#define ACC_TRIM_NVM_CTRL_PROG_MODE  0x01
```

```
// BMX055_ACC_BGW_SPI3_WDT     0x34
#define ACC_BGW_SPI3_WDT_I2C_WDT_EN  0x04
#define ACC_BGW_SPI3_WDT_I2C_WDT_SEL 0x02
#define ACC_BGW_SPI3_WDT_I2C_WDT_SPI3 0x01

// BMX055_ACC_Reserved         0x35

// BMX055_ACC_OFC_CTRL         0x36
#define ACC_OFC_CTRL_OFFSET_RESET  0x80
#define ACC_OFC_CTRL_CAL_TRIGGER  0x60
#define ACC_OFC_CTRL_CAL_READY   0x10
#define ACC_OFC_CTRL_HP_Z_EN   0x04
#define ACC_OFC_CTRL_HP_Y_EN   0x02
#define ACC_OFC_CTRL_HP_X_EN   0x01

// BMX055_ACC_OFC_SETTING      0x37
#define ACC_OFC_SETTING_OFFSET_TARGET_Z 0x60
#define ACC_OFC_SETTING_OFFSET_TARGET_Y 0x18
#define ACC_OFC_SETTING_OFFSET_TARGET_X 0x06
#define ACC_OFC_SETTING_CUT_OFF   0x01

// BMX055_ACC_OFC_OFFSET_X     0x38
#define ACC_OFC_OFFSET_OFFSET_X   0xFF

// BMX055_ACC_OFC_OFFSET_Y       0x39
#define ACC_OFC_OFFSET_OFFSET_Y   0xFF

// BMX055_ACC_OFC_OFFSET_Z     0x3A
#define ACC_OFC_OFFSET_OFFSET_Z   0xFF

// BMX055_ACC_TRIM_GP0         0x3B
#define ACC_TRIM_GP0_GP0       0xFF

// BMX055_ACC_TRIM_GP1         0x3C
#define ACC_TRIM_GP1_GP1       0xFF

// BMX055_ACC_Reserved         0x3D

// BMX055_ACC_FIFO_CONFIG_1      0x3E
#define ACC_FIFO_CONFIG_1_MODE   0xC0
#define ACC_FIFO_CONFIG_1_DATA_SELECT 0x03

// BMX055_ACC_FIFO_DATA        0x3F
#define ACC_FIFO_DATA_OUTPUT   0xFF
```

```
#endif
```

E.5 spi_imu.h file

```
#ifndef SPI_IMU_H
#define SPI_IMU_H


#define AK8963_ADDRESS 0x0C
#define WHO_AM_I_AK8963 0x00 // should return 0x48
#define INFO         0x01
#define AK8963_ST1   0x02 // data ready status bit 0
#define AK8963_XOUT_L 0x03 // data
#define AK8963_XOUT_H 0x04
#define AK8963_YOUT_L 0x05
#define AK8963_YOUT_H 0x06
#define AK8963_ZOUT_L 0x07
#define AK8963_ZOUT_H 0x08
#define AK8963_ST2   0x09 // Data overflow bit 3 and data read error status
    bit 2
#define AK8963_CNTL1  0x0A // Power down (0000), single-measurement (0001),
    self-test (1000) and Fuse ROM (1111) modes on bits 3:0
#define AK8963_CNTL2  0x0B
#define AK8963_ASTC  0x0C // Self test control
#define AK8963_I2CDIS 0x0F // I2C disable
#define AK8963_ASAX  0x10 // Fuse ROM x-axis sensitivity adjustment value
#define AK8963_ASAY  0x11 // Fuse ROM y-axis sensitivity adjustment value
#define AK8963_ASAZ  0x12 // Fuse ROM z-axis sensitivity adjustment value

#define SELF_TEST_X_GYRO 0x00
#define SELF_TEST_Y_GYRO 0x01
#define SELF_TEST_Z_GYRO 0x02

/*#define X_FINE_GAIN  0x03 // [7:0] fine gain
#define Y_FINE_GAIN  0x04
#define Z_FINE_GAIN  0x05
#define XA_OFFSET_H  0x06 // User-defined trim values for accelerometer
#define XA_OFFSET_L_TC 0x07
#define YA_OFFSET_H  0x08
#define YA_OFFSET_L_TC 0x09
#define ZA_OFFSET_H  0x0A
#define ZA_OFFSET_L_TC 0x0B */
```

```
#define SELF_TEST_X_ACCEL 0x0D
#define SELF_TEST_Y_ACCEL 0x0E
#define SELF_TEST_Z_ACCEL 0x0F

#define SELF_TEST_A  0x10

#define XG_OFFSET_H  0x13 // User-defined trim values for gyroscope
#define XG_OFFSET_L  0x14
#define YG_OFFSET_H  0x15
#define YG_OFFSET_L  0x16
#define ZG_OFFSET_H  0x17
#define ZG_OFFSET_L  0x18
#define SMPLRT_DIV   0x19
#define CONFIG       0x1A
#define GYRO_CONFIG  0x1B
#define ACCEL_CONFIG 0x1C
#define ACCEL_CONFIG2 0x1D
#define LP_ACCEL_ODR 0x1E
#define WOM_THR      0x1F

#define MOT_DUR      0x20 // Duration counter threshold for motion interrupt
    generation, 1 kHz rate, LSB = 1 ms
#define ZMOT_THR     0x21 // Zero-motion detection threshold bits [7:0]
#define ZRMOT_DUR    0x22 // Duration counter threshold for zero motion
    interrupt generation, 16 Hz rate, LSB = 64 ms

#define FIFO_EN      0x23
#define I2C_MST_CTRL 0x24
#define I2C_SLV0_ADDR 0x25
#define I2C_SLV0_REG 0x26
#define I2C_SLV0_CTRL 0x27
#define I2C_SLV1_ADDR 0x28
#define I2C_SLV1_REG 0x29
#define I2C_SLV1_CTRL 0x2A
#define I2C_SLV2_ADDR 0x2B
#define I2C_SLV2_REG 0x2C
#define I2C_SLV2_CTRL 0x2D
#define I2C_SLV3_ADDR 0x2E
#define I2C_SLV3_REG 0x2F
#define I2C_SLV3_CTRL 0x30
#define I2C_SLV4_ADDR 0x31
#define I2C_SLV4_REG 0x32
#define I2C_SLV4_DO  0x33
#define I2C_SLV4_CTRL 0x34
#define I2C_SLV4_DI  0x35
#define I2C_MST_STATUS 0x36
```

```
#define INT_PIN_CFG  0x37
#define INT_ENABLE   0x38
#define DMP_INT_STATUS 0x39 // Check DMP interrupt
#define INT_STATUS   0x3A
#define ACCEL_XOUT_H 0x3B
#define ACCEL_XOUT_L 0x3C
#define ACCEL_YOUT_H 0x3D
#define ACCEL_YOUT_L 0x3E
#define ACCEL_ZOUT_H 0x3F
#define ACCEL_ZOUT_L 0x40
#define TEMP_OUT_H   0x41
#define TEMP_OUT_L   0x42
#define GYRO_XOUT_H  0x43
#define GYRO_XOUT_L  0x44
#define GYRO_YOUT_H  0x45
#define GYRO_YOUT_L  0x46
#define GYRO_ZOUT_H  0x47
#define GYRO_ZOUT_L  0x48
#define EXT_SENS_DATA_00 0x49
#define EXT_SENS_DATA_01 0x4A
#define EXT_SENS_DATA_02 0x4B
#define EXT_SENS_DATA_03 0x4C
#define EXT_SENS_DATA_04 0x4D
#define EXT_SENS_DATA_05 0x4E
#define EXT_SENS_DATA_06 0x4F
#define EXT_SENS_DATA_07 0x50
#define EXT_SENS_DATA_08 0x51
#define EXT_SENS_DATA_09 0x52
#define EXT_SENS_DATA_10 0x53
#define EXT_SENS_DATA_11 0x54
#define EXT_SENS_DATA_12 0x55
#define EXT_SENS_DATA_13 0x56
#define EXT_SENS_DATA_14 0x57
#define EXT_SENS_DATA_15 0x58
#define EXT_SENS_DATA_16 0x59
#define EXT_SENS_DATA_17 0x5A
#define EXT_SENS_DATA_18 0x5B
#define EXT_SENS_DATA_19 0x5C
#define EXT_SENS_DATA_20 0x5D
#define EXT_SENS_DATA_21 0x5E
#define EXT_SENS_DATA_22 0x5F
#define EXT_SENS_DATA_23 0x60
#define MOT_DETECT_STATUS 0x61
#define I2C_SLV0_DO  0x63
#define I2C_SLV1_DO  0x64
#define I2C_SLV2_DO  0x65
#define I2C_SLV3_DO  0x66
```

```
#define I2C_MST_DELAY_CTRL 0x67
#define SIGNAL_PATH_RESET 0x68
#define MOT_DETECT_CTRL 0x69
#define USER_CTRL     0x6A // Bit 7 enable DMP, bit 3 reset DMP
#define PWR_MGMT_1    0x6B // Device defaults to the SLEEP mode
#define PWR_MGMT_2    0x6C
#define DMP_BANK      0x6D // Activates a specific bank in the DMP
#define DMP_RW_PNT    0x6E // Set read/write pointer to a specific start
    address in specified DMP bank
#define DMP_REG       0x6F // Register in DMP from which to read or to which
    to write
#define DMP_REG_1     0x70
#define DMP_REG_2     0x71
#define FIFO_COUNTH   0x72
#define FIFO_COUNTL   0x73
#define FIFO_R_W      0x74
#define WHO_AM_I_MPU9250 0x75 // Should return 0x71
#define XA_OFFSET_H   0x77
#define XA_OFFSET_L   0x78
#define YA_OFFSET_H   0x7A
#define YA_OFFSET_L   0x7B
#define ZA_OFFSET_H   0x7D
#define ZA_OFFSET_L   0x7E



#define READ_FLAG 0x80




#include <stdint.h>
#include "stm32f0xx.h"

/** pin-outs on KUKU sensor
  PB3: SCK
  PB4: MISO
  PB5: MOSI
  PA10: CS_bmx055_mag
  PA12: CS_bmx055_gyro
  PA15: CS_bmx055_accel
  PB6: CS_bmp280
  PA1: CS_uSD
**/

// select define
#define CS_BMX055_ACC_SELECT  GPIO_BSRR_BR_15
```

```
#define CS_BMX055_GYR_SELECT  GPIO_BSRR_BR_12
#define CS_BMX055_MAG_SELECT  GPIO_BSRR_BR_10
#define CS_BMP280_SELECT    GPIO_BSRR_BR_6

// deselect define
#define CS_BMX055_ACC_DESELECT  GPIO_BSRR_BS_15
#define CS_BMX055_GYR_DESELECT  GPIO_BSRR_BS_12
#define CS_BMX055_MAG_DESELECT  GPIO_BSRR_BS_10
#define CS_BMP280_DESELECT   GPIO_BSRR_BS_6

/**
  * @brief Initializes the SPI/IMUBP communication.
  * @param None
  * @retval None
  */
void init_spi_ssm(void);

/**
  * @brief Writes a block on the BMX055/BMP280
  * @param address: register address to access
  * @param data: data to write on the register
  * @param select: chip select
  * @param deselect: chip deselect
  * @retval None
  */
void write_to_address(uint16_t address, uint8_t data, uint32_t select,
    uint32_t deselect);

/**
  * @brief Reads a block on the BMX055/BMP280
  * @param address: register address to access
  * @param data: data to read from the register
  * @param enble: chip select BSRR u32
  * @param deselect: chip deselect BSRR u32
  * @retval data
  */
uint8_t read_from_address(uint16_t address, uint32_t select, uint32_t
    deselect);

#endif
```

F Further testing

F.1 Size range of obstacles

F.1.1 Test procedure for test F.1

Aim: To determine the robots ability to overcome obstacles of varying sizes
Justification:A test of how well the design will be able to overcome objects of a certain size is imperative as it is a crucial requirement of the design. It is also important to highlight the robots abilities as well as its limitations in order to make improvements.
procedure:This test can be done by placing different sized obstacles of wood (or concrete if possible) in the path of the robot and using any maneuvers possible (including full speed different approach angles or any other means necessary) to get the robot to successfully overcome the obstacle. A log should be kept of the results.

F.1.2 Results for test F.1

The results of the test to determine the ability of the robot to overcome various size objects can be seen in table **??**

Table F.1: Log for ability to overcome various sized obstacles

Obstacle size [mm]	Success (Yes/No)	Short description of best method
100		
...		
200		

F.2 Ability to climb at varying speeds

F.2.1 Test procedure for test F.2

Aim: To determine the robots ability to overcome obstacles at varying speeds
Justification: It is important to know the speed at which the robot is best capable to overcome obstacles in order to provide the operator with some form of strategy for success. Knowing how the speed affects climbing will also provide useful information for improvements.
Procedure: This test will explore the possibility that robots likelihood of success in

overcoming objects may depend on the robots approach speed and/or the speed being applied while in contact with the surface.
In order to reliably test this simply estimating how far forward the joysticks on the controller are pressed will not be accurate or repeatable. In order to ensure consistent repeatable results the code on the micro controller will need to be changed between each test. The maximum speed of the robot will need to be set (by changing the multiplier between the signal received from the remote and the PWM output to the H-Bridge) and the robot will need to be driven with the joysticks in the full throttle position. A log should be kept of the speed (as a percentage of full speed) ,whether the robot had a run up or if it started next to the obstacle and whether the robot was successful in overcoming the obstacle. This test can be repeated with various obstacle surfaces and heights to see if there is a variation in which speed is most effective with variations in material and height. This test aims to provide data as to what approach is best for an operator to take when overcoming an obstacle and to provide information as to what improvements can be made to the design.

F.2.2 Results for test F.2

The results of the test to determine the effect of speed on ability to climb can be seen in table F.2

Table F.2: Log for effect of speed on success in overcoming obstacles

Speed percentage	Run up (Yes/No)	Obstacle Height	Obstacle material	Outcome
10				
20				
...				
100				

F.3 Effect of angle of wheel contact

F.3.1 Test procedure for test F.3

Aim: To determine the robots ability to overcome obstacles at different approach angles.
Justification: The angle of wheel contact with an obstacle or henceforth referred to as the angle of attack) may play an important role in the success in overcoming the object. It is important to know whether it will make a difference in order to approach obstacles with the best chance of success and in order to improve any shortcomings.
Procedure: The effect of the angle of attack can be tested methodically and repeatably by placing an obstacle on the floor and marking off lines leading up to the obstacle at different angles, the robot can be driven up to the obstacle along these specific lines and

the success of the robot should be recorded in a log along with a brief description of what occurred. These angles must be measured with reference to he front plane of the obstacle where a head on climb would constitute a 90^o attempt.

F.3.2 Results for test F.3

The results of the test to determine the effect of angle of attack on the robots ability to overcome obstacles can be seen in table F.3

Table F.3: Log for angle of attack test

Angle (^{o})	Success (Yes/No)	Description of what happened Description of what happened
30		
40		