University of New South Wales

ENGG1000 Team Ten
Mars Regolith Collection
Engineering Design Report

ANTHONY DO
COLINE LU
CODY MCCARNEY
DANIEL PADOANI
DANIEL SHARP
CHRISTOPHER TSANG

27-04-2020

Contents

Summary

Future Martian explorers will require protection from radiation. One potential solution is to mine Mars regolith and 3D print radiation shelters from it. This project explores the design and fabrication of a prototype regolith collection rover. A problem statement was generated. Design features were analysed and a prototype designed in CAD. The prototype was constructed and evaluated as a solution

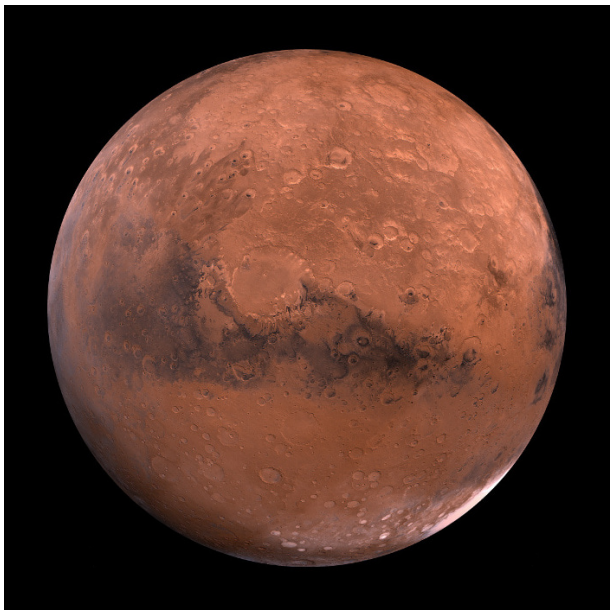


Figure 1: Mars.

Acknowledgements

We would like to acknowledge the support of Mitchell Torok our team mentor. We are also grateful to Binghao Li and the School of Minerals and Energy Resources for guidance and advice over the course of this project.

1. Problem Definition and Background

1.1 Background

1.1.1 Setting the scene

The exploration of Mars has taken place over hundreds of years. Numerous probes, satellites and several rovers were sent to the Red Planet for geological investigations, habitability potential,surface exploration,life existence and natural resources research.The latest success is the landing of InSight probe on Mars on Nov. 26, 2018.

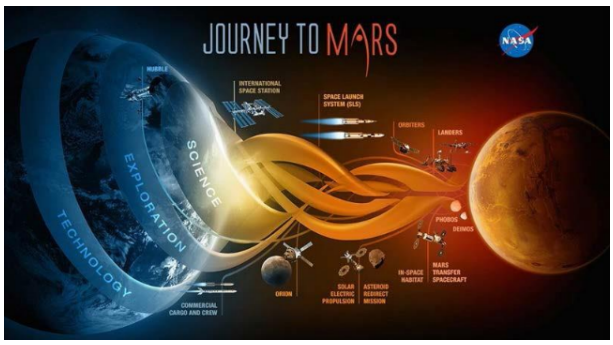


Figure 1.1: NASA's Mars Exploration Road Map

Many people have long advocated a manned mission to Mars as the next logical step for a manned space program after lunar exploration. Fifteen years ago, this idea was hardly taken seriously. Today however, there are real projects the leading space companies are working on. For example, The European Space Agency, ESA[5] has plans to land humans on Mars between 2030 and 2035. Manned exploration by the United States was identified as a long-term goal in the Vision for Space Exploration announced in 2004. The Orion spacecraft, successfully launched in 2014, is planned to be used to send a human expedition to our Earth's moon by 2020 as a stepping stone to a Mars expedition as NASA aims to put a person on Mars by 2037

There are numerous problems an initial habitant will face once they have landed on Mars. One of these difficulties is the high radiation exposure. New spacesuits and spacecraft are to be designed and built before a Mars walk will be safe for a human being. This project examines a later stage – the building of a radiation protection shelter for a Mars colony or resource storage. As it is expensive to deliver building construction materials from Earth, it seems more efficient to use materials sourced locally. As a terrestrial planet, Mars consists of minerals containing silicon and oxygen, metals and other elements that typically makeup rock. This project requires the design, construction and testing of a machine that can drive on a "Mars-like surface", collect/mine resource and bring it to the container for resource shelter building for radiation protection.[7]

1.1.2 Project Design Objective

The project objective states:

" This project requires the design, construction and testing of an item of equipment that can drive on a "Mars-like surface", collect resource and transport it to the feeder of a 3D printer on "Mars". Mined resources will be used to print a shelter/radiation protection infrastructure for Mars colonisation. We will provide you with a map of the Martian surface your rover is to traverse, so you can plan your resource collection strategy. How you are going to collect the resource and deliver it to the feeder is up to you.You may need to do some filtering out of larger items if crushing may be a challenge? However shelter building needs a lot of resource - the more – the better...The objective of this project is for your team to design and build a prototype rover that is able to transit on a "Mars-like surface" and collect samples of the resource. As are representative test bed we will use a beach terrain with small rocks, sand hills and recesses(see Figure 2). The test area measures $9m^2$,equipped with a container for collected resource.[7]"

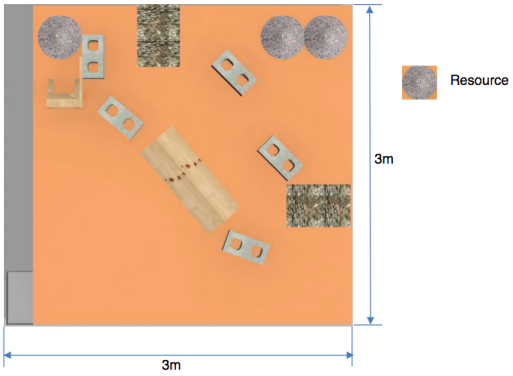


Figure 1.2: Test Area

Figure 1.3: Regolith Advanced Surface Systems Operations Robot (RASSOR) Excavator.

1.2 Research

The first step in designing our solution was research. We started with general Mars exploration and earth based mining. As we gained better understanding of the project we focused our research. We did a lot of reading on small scale robot design, construction and programming. As the design evolved and as the design evolved during the build we returned to the library many times.

As a primer we read two general Mar engineering books. 'The Right Kind of Crazy'[14] and "Mars Rover Curiosity: An Inside Account from Curiosity's Chief Engineer"[8]. Both are about the design and engineering challenges of building and landing the Curiosity rover.

We used YouTube to research NASA's Lunar Mining competition for ideas and features. We also watched videos of open pit mining machines on earth to see what designs are prevalent.

- NASA's off-world mining solution: RASSOR.[3]
- Lunarbotics competitions[9]
- Bucket wheel open pit mining)[4]
- Excavator mining.[1].

Our mentor whose team won the previous years MERE project was our best source of information. He was able to give extremely focused advice throughout the project. Most importantly, he advised us to start building early and keep the design simple. The project is not easy and most teams fail to produce a working design.

We were also shown video and pictures of previous designs. We analysed these for common features and tried to analyse they were selected been chosen.

Figure 1.4: A previous project utilizing a rotating 'bucket' and conveyor belt.

Figure 1.5: A previous project using a scoop as the collection and transportation device.

As the project progressed we needed specific information on small scale radio controlled robots. An advantage of using the Arduino microprocessor is thousands of projects are documented on Github and other websites. In particular we relied on mobilerobots and rcarduino for ideas and inspiration. For practical technical information we used two main references. 'Practical Electronics for Inventors'[12] and 'Arduino Robotics'[6]. Both of these books are excellent. Having access to them was a major factor in the success of the project. In our research meetings we polled team members for resources they already owned or could access.

Reading the books about Curiosity gave us an appreciation of the phrase 'Space is hard'. From the initial research phase we concluded that the ten week time limit was the major constraint. A simple design would be the easiest to construct and test within the time-frame. We decide to adopt features we saw successfully used in previous years to reduce development time. We observed the most successful designs had enough capacity and power to carry a large amount of regolith. The weaker designs barely powerful enough to carry more than a few pieces. We resolved to avoid this mistake. From our team poll we concluded we had a radio control transmitter and 24V Lipo batteries to control and power our design. This would reduce our costs and allow a bigger budget for materials.

2. Problem Statement Generation

2.1 Considerations

To generate our problem statement we considered the following elements.

- **Who** A group of first year engineering students. The client is 'NASA' but in relation to the actual problem the client is the course assessor and our team as we will be the ones operating it.
- **What** Design and build a prototype rover that is able to transit on a Mars-like surface and collect resources.
- **Constraints**
    1. Limited skills - it must be built by students with little or no experience in building robots.
    2. Limited time: - a working prototype must be produced in ten weeks.
    3. Limited operating area - It must also operate in a $3 * 3m$ environment so must be small enough to manoeuvre in this space but also large enough to carry approx. 1 kg to 5kg of regolith. More is better.
    4. Limited operating time - the rover has four minutes to collect the regolith.

2.2 Problem Statement

After considering the above we generated the following problem statement;

"As part of their assessment for a first-year engineering program students are required to produce a working prototype of a rover within approximately 10 weeks.

The rover needs to be able to transit through a Mars-like surface, simulated by the use of small rocks, sand hills and recesses and collect resources, which are simulated by small pebbles. It must operate continuously for 4 minutes and traverse an area approximately $9m^2$.

The assessment will take into account whether a prototype is produced and the rate of regolith collection. Marks will also be given for energy efficiency and innovation."

2.3 Design Principles

From the problem statement we identified that our **key issues** were limited time and limited skills. The **actions** we took to mitigate these issues were to formulate two design principles.
These were;

1. **Simple: easy to design, build and operate.**
    - made from readily available materials that don't require special tools.
    - The simplest design possible with the least number of parts.
    - Use as much 'off the shelf' electronics as possible to minimize development time.
    - Make use of resources we have already to minimize expense.
2. **Powerful and large enough to move between 1 - 5kg of regolith in four minutes.**
    - Use high voltage. 12 or 24 volts.
    - Be large enough to carry between one and five kilograms of regolith at one time.

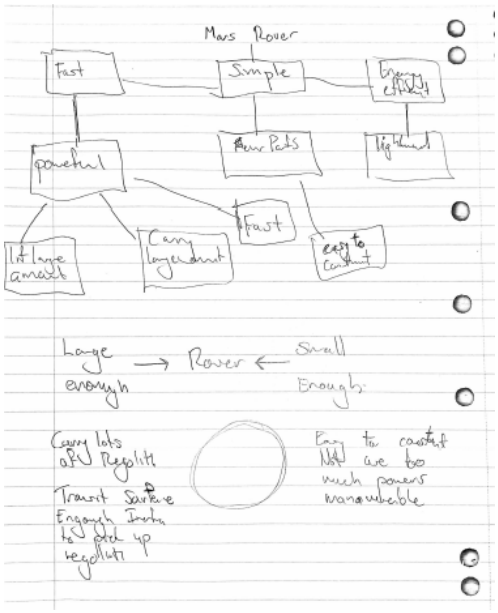


Figure 2.1: Drafting design principles.

3. Design Concept Generation

3.1 Concept Sketches and Inspiration

We generated design concepts incorporating ideas and features from our research. Each concept cossists of three systems. These need to work together. These are: the Regolith collecting system, the method of movement and steering, and the method of storing and dispatching the regolith. Each team members design concepts were sketched up and their merits analysed against our design principles.

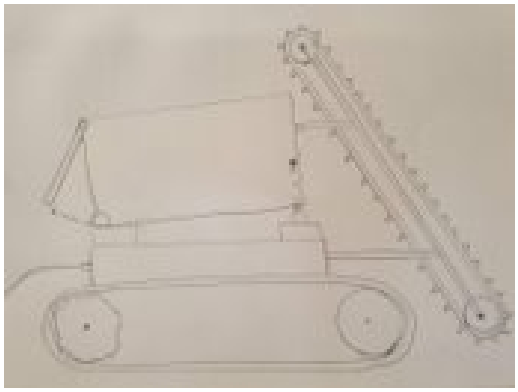


Figure 3.1: Tracked with small buckets on conveyor belt concept sketch.

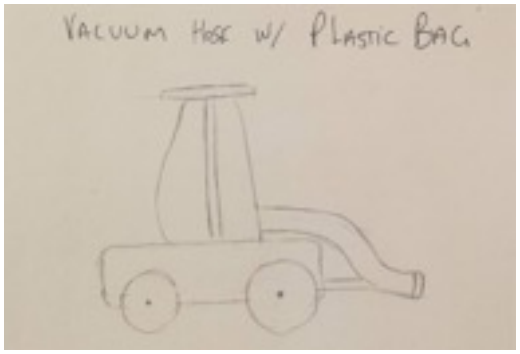


Figure 3.2: Vacuum mounted on wheel concept sketch.

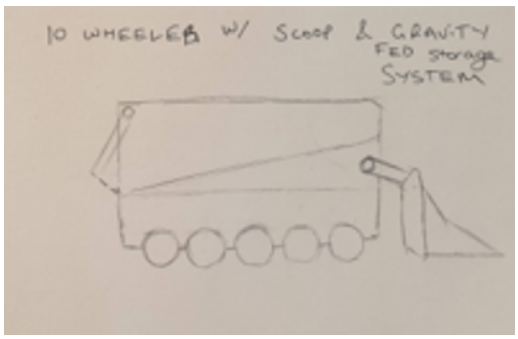


Figure 3.3: Multiple wheeled with scoop concept sketch.

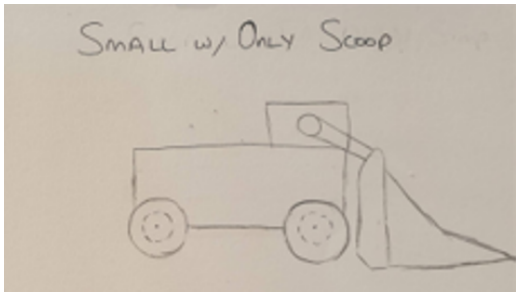


Figure 3.4: Small with regolith transported in scoop concept sketch.

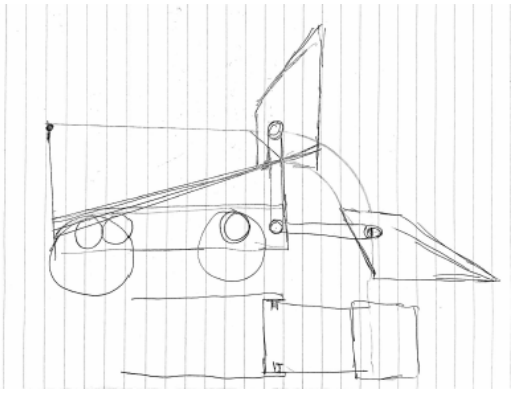


Figure 3.5: Scoop with gravity fed regolith container.

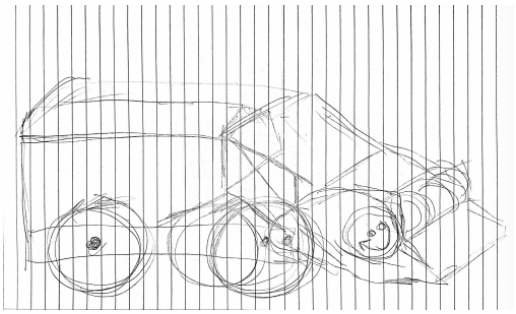


Figure 3.6: Horizontal sweeping roller concept sketch.

3.1.1 Analysis of Design Concepts

Dimensions and Regolith Capacity We analysed the map of the test area in figure 2.2. Some regolith is located closer to to the dispatch area, but in a 'cave'. We decided to ignore the 'cave' regolith and dimension the rover to carry and collect as much as possible from the more accessible piles whilst still be small enough to manoeuvre around obstacles. We decided on a length of between 30 and 50 cm and a width of not more than 30cm. To be able to contest the competition record we aimed for design load of between 1 and 5kg.

Movement Initially we favoured using tracks over wheels. We thought tracks would provide better traction and be easier to implement. This is because only one motor would be need for each side. After researching track mechanisms and current Mars rover designs we decided on wheels. RC truck wheels of various sizes are readily available and relatively inexpensive. Track systems are expensive to buy and complex to build. We also considered using a drone with a claw system attached, but rejected this idea as drones are also very complex.

Regolith collection The method of collecting the regolith provoked the most debate. The analysis was complicated by a sample not being available. Therefore we had to consider the ability of each method to handle regolith of many different scales. This was the main factor in selecting a scoop design. A vacuumn cleaner' approach was also considered. This is a great idea in theory, but falls apart on examination. Vacuum cleaners do not work on Mars.
A rotating bucket was considered, but rejected because of the complexity. The designs considered viable were a scoop or a horizontal sweeping roller. We decided a scoop had the most potential. A scoop design is simple and can handle a wide variety of potential regolith sizes.

Regolith storage and dispatch Two concepts were considered. A triangular box with a sloping floor to dispatch the regolith by gravity or not having a container at all and carrying regolith in the collection device. Although not having a separate storage container is the simplest option it would not allow a high rate of collection.

3.2 Morph Chart and Design Feature Selection

3.2.1 Morph Chart Generation

The most viable solutions from the concept analysis were used to generate a number of potential solutions. These were drafted into a Morph Chart matrix. In a group meeting each feature was analyzed for its compatibility with our design principles. A discussion of the morph chart features and the reasons for selection follow:

3.2.2 Morph Chart

Function / Feature	Option 1	Option 2	Option 3	Option 4
Body Material	Plywood	Aluminum Sheet	Fiberglass	Carbon Fiber
Fabrication Method (subsystem of material)	Nuts & Bolts	Glue (Titebond or JB Weld)	Screws	
Power Source	Batteries	Fuel Cell	Solar Panels	Internal combustion
Battery Type (Power source subsystem)	Li-po	Lead-Acid	Ni-Cad	
Voltage (Battery subsystem)	24v	12v	9v	5v
Propulsion (power source subsystem)	DC Electric Motors (Direct Drive)	DC Electric Motors (Indirect Drive)		
Movement	Wheels (Four-Wheel Drive)	Wheels (Two-Wheel Drive)	Tracks	
Steering	Differential steering	Two moveable wheels	Four moveable wheels	
Control Method (Microprocessor)	Arduino	Raspberry Pi	STMicroelectronics NUCLEO-F401RE	
Control Method (Communication)	Radio Control 2.4 GH RC transmitter and receiver	Bluetooth	Wi-Fi	Autonomous
Regolith Collection	Scoop	Horizontal sweeping roller	Vacuum System	Rotating Drum with scoops
Scoop Actuator (Regolith Collection sub-system)	Linear actuator	Servo	Pneumatic/Hydraulic	
Regolith Dispatching System	Gravity fed with gravity opened gate.	"Dump-Truck" – tipping tray.		
Gate Actuator (Regolith Dispatching sub-system)	Solenoid	Linear actuator	Servo	

Figure 3.7: Morph Chart: green indicates selected features

3.2.3 Body Material

Plywood was the only serious contender. It is light, strong, cheap and easy to work. We used 7mm ply for the main body and 3mm ply for the scoop and rear gate.

In the age of composites, plywood seems like a backwards choice. However, plywood has many of the advantages of composites and almost none of the disadvantages. See A brief history of Plywood for details.[13]

3.2.4 Fabrication Method

We used machine nuts and bolts in combination with angle brackets to assemble the frame. Being able to easily assemble and disassemble is one of the most importance aspects of our design. Inevitable issues and technical problems arose during the build solutions are much easier to develop if the design is adaptable. Some of these issues can be avoided by the use of CAD and simulation but built in flexibility is crucial. It also improves ease of maintenance once operational.

3.2.5 Power Source

We had a number of 24 volt 6 cell 1450 mAh Li-Po batteries. We used these because Li-Po's are energy dense, light weight and expensive. Not having to buy batteries reduced costs. See Li-Po Batteries[11] for more information.

3.2.6 Battery Type and Voltage

Battery type and voltage were dictated by our choice of Li-Po's. Using a higher voltage means the current draw is lower. This means lighter gauge wire can be used and heating losses reduced. It also means that digital systems running from the main battery can maintain a steady 5V supply even under significant voltage drop from start-up or stall loads.

3.2.7 Propulsion

The propulsion system is electric as dictated by our power source. We chose direct drive from geared down 24V motors with a RPM of 120. Direct drive saves space and reduces complexity as there are no external gearboxes or driveshafts.

3.2.8 Movement

Here we chosen to accept a slight increase in complexity. Four-wheel drive needs the addition of two motors and draws twice as much current, but gives a large increase in the ability of the rover to deal with sand and carry more regolith. Our system is based on the robot detailed here.[10]

3.2.9 Steering

Skid steering was the obvious choice. Skid steer can be implemented easily as the wheels don't have to move in the vertical plane. Skid steer is software based so infinitely adjustable to suit the operating conditions.

3.2.10 Microprocessor

Arduino was the obvious choice. There are development boards with higher specs. For example, Raspberry Pi and Nucleo. However, the amount of readily available documentation, support, libraries and open source code made this choice for us. We used the Arduino Mega to maximise the number of PWM outputs whohc we needed for motor speed control.

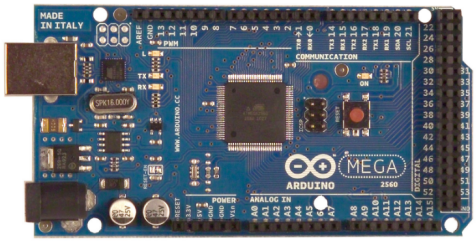


Figure 3.8: Arduino Mega 2560

Figure 3.9: The drive train of our rover is based on this design by mobilerobots.pl.

3.2.11 Control Method

We had a 7 Channel 2.4 GH Radio Control transmitter already available. This meant a slight increase in complexity because we need to code the Arduino to process the PWM signals from the receiver. This is acceptable because a transmitter is designed for precise control. The rover is much easier to operate than if using a laptop. There is also a large amount of information on RC signal processing with Arduino. See RC Arduino[2]

3.2.12 Regolith Collection

Initially we were in favour of a horizontal sweeping roller to collect the regolith as we felt it would offer a greater rate of collection. However, after the problem statement generation exercise and discussions with the team we decided to go with a scoop. The reasons for this were;

1. In all the previous rovers that we saw, no solution included a roller even though we know a lot of teams considered it. This made us think it was harder to implement than it seems.
2. We haven't seen a sample of the Regolith yet. It may or may not be suitable for a sweeping roller system. A scoop can deal with many different types of Regolith, but a roller would be more likely to need customisation to operate with a particular 'spec'.
3. A sweeping roller would need a system to raise and lower the roller arm and a system to spin the roller adding complexity. We are aiming for the simplest solution possible.
4. Scoop are a 'tried and true' system. We know from the many examples in earth moving and mining machines and the almost exclusive use of them by previous teams.

3.2.13 Scoop Actuator

We have chosen a linear actuator over servos for three reasons.

1. Using a Linear actuator lets us apply force to the scoop arm at a distance from the hinge reducing the load on the system.
2. Most readily available servos run on 5 – 7.5 Volts. We want to use a standard voltage for the whole system to reduce complexity.
3. 24V linear actuators are relatively cheap compared to 24V servos.

3.2.14 Regolith Dispatching System

We made a compromise towards simplicity at a reduction in load capacity. A gravity fed system needs no moving parts but requires a triangular profile that cuts the useful volume in half. The gate that closes under gravity and opens under the weight of the regolith. Therefore, the only electro-mechanical system need is a device to keep the gate closed under load and allow it to open at the dispatching area. The 'area under the triangle' is utilised for the battery and electronics so it is not wasted.

3.2.15 Gate Actuator

To allow the gate to open and close we planned to use a 24 Volt solenoid as a latch.

3.3 CAD Design

We designed the rover in Fusion 360 following the design principles and features outlined above. To save time only the basic structure and dimensions were designed in CAD. The fabrication brackets and smaller components were not modelled.

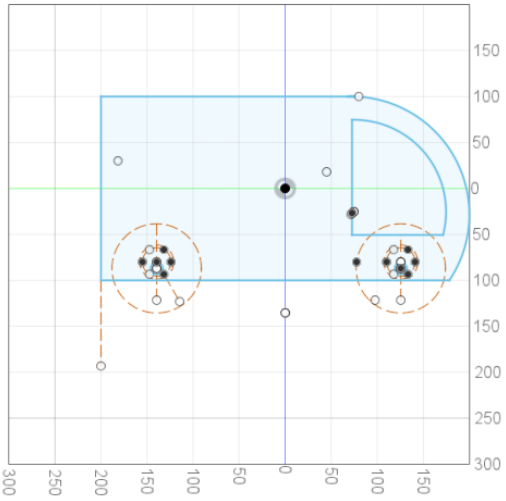


Figure 3.10: Main body profile sketch.

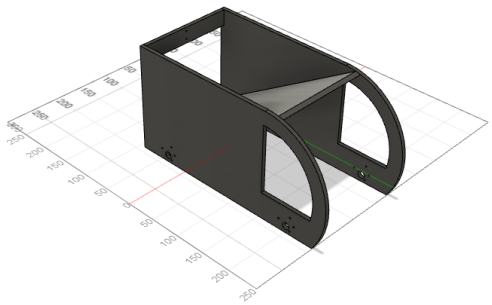


Figure 3.11: Main body in Fusion 360.

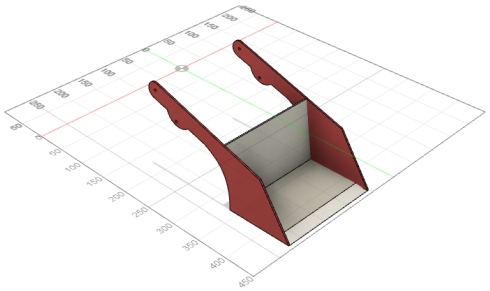


Figure 3.12: Scoop in Fusion 360.

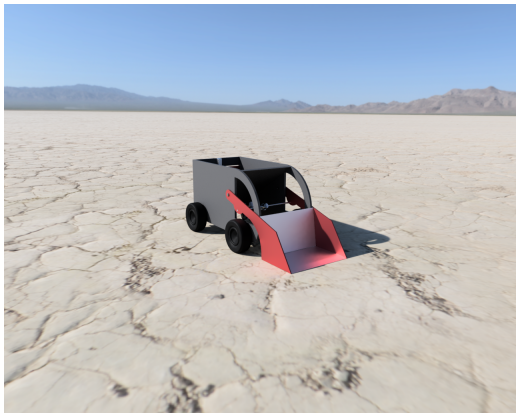


Figure 3.13: Fusion 360 render of rover on mars.

The design is a box shape with an angled 'floor' running most of the length to form the container and provide structural support. There is a cross brace at rear to form a rigid triangular support structure.

The motors are bolted directly to the side panels. The linear actuator is supported by 8mm diameter stainless steel rod and located in the centre. The rear gate hinge is offset so it closes under its own weight. The solenoid is located on the top brace and prevents the gate from opening.

If we had been more familiar with Fusion 360 we could have used some of the more advanced features such as movable joints and simulation. The use of CAD was a great help in analysing the geometry of the scoop and positioning the motors.

4. Rover Build Process

4.1 Mechanical Assembly

4.1.1 Sourcing Components

As per our design principles our rover was constructed of readily available materials. All components were purchased from ebay, Facebook Marketplace, Bunnings and Jaycar Electronics. A complete purchasing list is included in Appendix D.

4.1.2 Laser Cutting

A key reason for using CAD was the design profiles can be used to laser cut the plywood frames.

A video of laser cutting the scoop arms is here.

4.1.3 Materials and Method of Construction

The main body was constructed of 7mm interior plywood. The scoop was initially constructed from 3mm ply. We used galvanised right-angle brackets and M3 bolts and nuts with flat washers. The use of nuts and bolts was important as it gave our rover the ability to be dismantled and reassembled easily. The linear actuator pivots, rear gate pivots were constructed from stainless steel rod.

Figure 4.1: Secondhand $\frac{1}{10}$ Scale RC Off-Road Truck Wheels.

Figure 4.2: Main body and scoop assembled.

Video of body and scoop assembled.

4.2 Electro-Mechanical Assembly

4.2.1 Scoop Redesign

In contrast to the construction of the rover main body, the construction and integration of the electro-mechanical systems had many issues and required a major redesign of the scoop mechanism, rear gate and electronics layout.

4.2.2 Linear Actuator Issues

These issues stemmed from trying to save on costs by purchasing a non-branded cheap linear actuator. The actuator sourced from ebay is approximately five times cheaper than a brand name item. No data sheet was online and the vendor could not obtain one. This meant when the CAD model was designed, we underestimated the actuator dimensions and weight by a factor of two. The geometry of the scoop operation had to be redesigned to fit a much larger and heavier actuator than expected.

To fit the large actuator, we moved the attachment of the scoop pivot from a metal rod between the scoop arms, to attach directly to the scoop. We also re-fabricated the floor and back of the scoop in sheet metal because it was flexing under the load from the actuator. We also changed the diameter of the actuator pivot from 6mm diameter stainless steel rod to 8mm to eliminate some bending.

We also needed to remove the front brace and move the Arduino to under the regolith container. This redesign was ultimately better becuase it reduced the number of parts and lowered the center of gravity. However, the Arduino was harder to access.

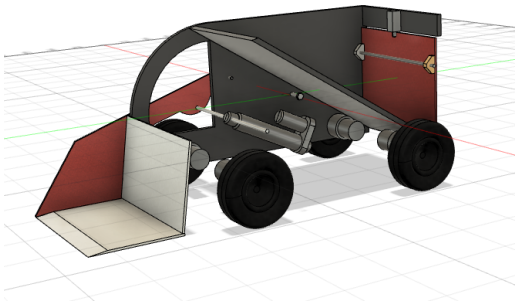


Figure 4.3: Actuator and scoop as designed in CAD.

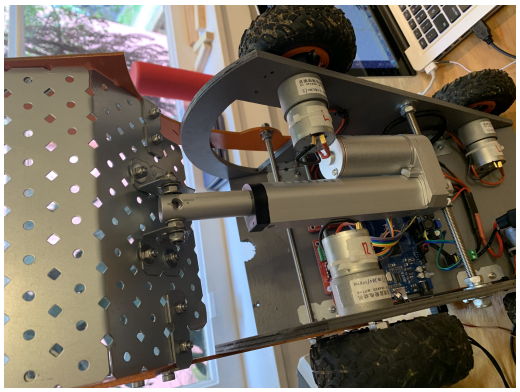


Figure 4.4: Actuator and scoop after re-design

4.2.3 Rear Gate Control Issues

The original design controlled the rear gate with a 24V solenoid (see fig 5.7). Unfortunately very late in the build period the eBay vendor canceled the order. With the COVID-19 quarantine in full effect we could not purchase another. The design was adapted to use two Gaui GS-311 metal gear servos to actively hold the gate closed. This also required adjusting the Arduino power from 12 V to 6V as the servos are powered directly from the battery. This design worked well, but because it was implemented at a very late stage, looks out of place. We may replace these with a solenoid if a second iteration is built.

A video of a rear gate servo operation test is here.

4.2.4 Regolith Sample and Testing

We were provided with a regolith sample in week four. It would have been advantageous to have this sample before the design was completed but we could not afford to delay construction due to time constraints.

The 'regolith' is angular and has a lot of flat surfaces. It has a high angle of repose (approximately 40 degrees) and does not slide easily. This is difficult for gravity fed systems to handle and we had to adjust the angle of the scoop back to allow it to slide. To lower the friction co-efficient we constructed a Corflute inner on the scoop and regolith container. This worked well and is also easily replaceable if damaged.

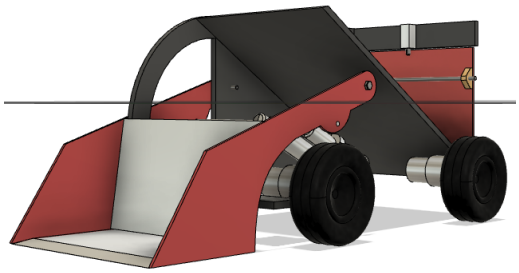


Figure 4.5: Rear gate and solenoid as designed in CAD.

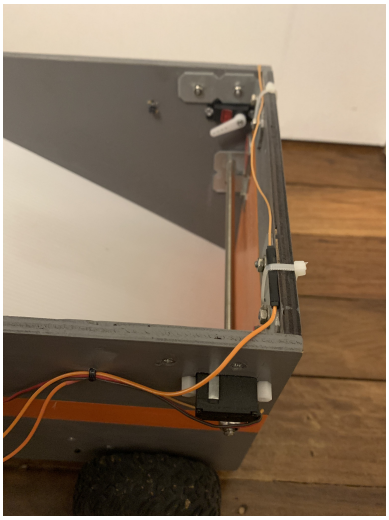


Figure 4.6: Rear gate control servos after redesign

Figure 4.7: A 'Regolith' Sample

4.3 Electronics and Control System

4.3.1 Implementation

To implement the electronic system we considered two options:

1. Pre-constructed Arduino compatible modules wired together
    - Pros – simplest option, quick to implement, easy to test and likely to work first time.
    - Cons – takes up lots of space and requires more wiring, parts of some modules are not used. If something goes wrong, it may take more time to figure out the issue.
2. Design a separate PCB with Arduino, DC power step-down, motor controllers on one board.
    - Pros – compact and uses less power, no redundant components, the team knows exactly how it works.
    - Cons – takes much more time to implement

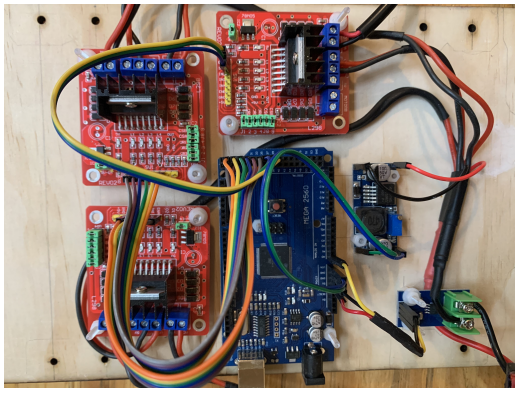


Figure 4.8: Electronics Layout.

Figure 4.9: Complete electronics and mechatronic system removed from the rover for testing.

We chose to use modules designed for the Arduino due to time constraints and lack of access to PCB manufacturing facilities. If we continue to develop the design in the future, we would like to design a custom PCB to make more space available for other sensors.

The electronics system has seven components:

- A receiver to receive signals from the transmitter.
- Arduino Mega microprocessor to process the receiver inputs and send signals to the motor controllers.
- Three L298N Dual Motor Drivers – Left and right motors have one each and one to drive the linear actuator. The actuator driver has one redundant input/output.
- A DC – DC converter to provide power to the Arduino and rear gate servos from the battery.
- An ACS712 Hall-Effect Current Sensor to measure the system current draw.

Data sheets for the major electronic modules can be found in Appendix A.

Most of the work involved constructing the wiring loom. A lot of time was spent making the loom reliable and unlikely to be compromised by vibration. We used heat shrink insulation around vulnerable areas and nylon fastenings to mount the modules.

The operation of Arduino, DC MOSFET motor controllers and PWM signals is well documented in many resources so we will not re-explain it. A detailed explanation can be found in Monk et al. (2016) in chapters 13 and 15.[12]

One feature we should have included but did not is a fuse. During testing we shorted out the battery wires resulting in a small but spectacular fire as the current sensor acted as an involuntary and expensive one way circuit breaker. This incident could have resulted in a serious battery fire and a fuse will definitely be included in the next iteration of the rover.

4.4 Final Assembly

4.4.1 Colour Scheme

Figure 4.10: Inspired by Ridley Scott's 'The Martian'.

Figure 4.11: Painting the components.

When we were satisfied that the systems were working we disassembled and painted the main body components. This was done to increase the aesthetic appeal of the rover. Vision is one of the most dominant senses in humans. A design that is visually appealing is more likely to be selected. We chose an orange and grey colour scheme as a reference to Riddley Scott's 'The Martian'.

Final assembly was carried out using Loctite on all nuts and bolts to reduce the chance of vibration induced failures.

The linear actuator and rear gate pivots were lubricated with a lithium based dry lube.

The battery was attached to the rover with zip ties. These add no weight and hold the battery securely.

5. Arduino Code Development

5.1 Code Development and Testing

5.1.1 Code Development Environment

The rover has an USB port to enable the IDE to be plugged into the Arduino and code tested immediately. This was a great help in developing the code quickly. We were also able to monitor the current draw and RC inputs to the Arduino with the serial plotter in real time.

Figure 5.1: Serial monitor plot of throttle and steering channel inputs.

5.1.2 Resources

The code for this project is based on and developed from the ExplorerBot code in Arduino Robotics[6].

The complete test and operating programs are included in Appendix B.

5.1.3 Motor Controller Issues

The motor controllers were tested with test code that had no RC input. This allowed us to test the Arduino and motor controllers and become familiar with the system before programming the operating code.

One issue with the linear actuator motor controller was identified during testing. The controller had power and the Arduino was switching the inputs correctly, however the controller was not supplying power to the motor output. After independent testing we discovered an intermittent cable fault. The cable tested fine but when re-installed was not sending the signal to the controller. This did not show up on the controller as the switching LED's do not required the enable signal to illuminate. We should have used the oscilloscope instead of relying on the LED's.

A video of the motor demo test code running.

5.2 Operating Code

5.2.1 General Principles

The idea behind the control system is simple. The RC transmitter sends PWM signals to the receiver. These inputs are processed by the Arduino and output as signals to the motor controllers to steer the rover and operate the scoop.

Figure 5.2: The transmitter. The right stick y-axis controls forward and reverse. The x-axis controls left and right. The left stick y-axis controls the scoop.

We analyzed the output from the receivers to determine the centre and minimum and maximum value of each channel.

A video of this process is here.

The PWM frequency of the Arduino outputs was changed to $32kHz$. This raises the frequency of the PWM signal above the threshold of human hearing so the rover is quieter.

The main part of code is a series of 'nested if' loops. This loop constantly checks the position of the control sticks and changes the output signal to the motor controllers in response.

The skid steering is implemented by slowing the 'inside' wheels proportionally to the input from the steering channel. If there is no input from the forward/reverse channel the 'inside' wheels will reverse to spin the rover on the spot.

The scoop control is a simple on/off code. If the channel value is negative the scoop will be raised. It is it positive the scoop is lowered. There is a 'deadband' when the stick is centred to stop the scoop at a given position.

The servo's that keep the rear gate shut are controlled directly from the receiver because the PWM signal does not need to be processed.

The complete code which is well commented in included in appendix B.

6. Testing

6.1 Testing Principles

During the build and development process we tested each system as soon it was operating. The idea was to eliminate issues at an early stage. This was successful as the rover only required one small alteration once completed.

6.2 Scoop and Linear Actuator

The linear actuator and scoop system required the most extensive separate testing. As mentioned above, the scoop was redesigned after the original wooden design was flexing. A lot of testing and retesting was required to balance the range of motion of the actuator with the angle of the scoop. Because the geometry of the design had been changed, we were not able to raise the scoop to the degree that was originally planned. This meant the regolith would not slide off into the storage container and was a serious problem. As mentioned above this was solved by use of a Corflute inner to adjust the angle of the rear face of the scoop.

6.3 Integrated Testing

The separate system tests meant an extensive integrated testing phase was unnecessary. One small modification was needed after integrated testing. We added rubber pads to eliminate a small gap allowing regolith to become stuck and prevent closing.

Figure 6.1: Completed rover ready for testing.

Figure 6.2: Rubber pads added to rear gate assembly.

Because of the COVID-19 quarantine conditions we were unable to compete against other teams. In lieu of the competition we designed three tests. Test A is recreation of the competition which tests the collection rate, energy efficiency and overall operation. Test B is a test to calculate the maximum operating speed. Test C determines the maxiumum load.

To measure energy use we used a programmable charger that records the mAh required to recharge. By recharging immediately after the test we hoped to measure energy use accurately.

6.3.1 Test A

Aim: To determine the regolith collection rate and energy efficiency. We also evaluated the driving experience and general effectiveness of the design.

Method: A sand and gravel driveway with a gentle slope was used to simulate a 'Mars-like surface". Two 4kg piles of gravel chip were setup 4.25 metres from the regolith collection box. To simulate an obstacle the collection box was reached via a ramp. See figures 7.3 and 7.4 below.

Starting from the collection area the rover collected as much regolith as possible from the piles and deposited it in the collection area. The collected regolith was weighed and the batteries recharged.

The average collection rate was determined by dividing the total amount of regolith collected in both four minute tests by the total time in seconds. This was converted to kilograms per hour.

We took care to only use each battery for the duration of the test. Each battery was recharged immediately after the test and mAh put back into the pack recorded.

The energy used was calculated by the multiplying the average mAh recorded by the battery voltage. This product was multiplied by .015 to get the energy use per hour in Watt hours.

Figure 6.3: The 'regolith'

Figure 6.4: Regolith collection box.

A time-lapse of the first run in this test is here.

Results and Calculations:

Table 6.1: Test A. Regolith collection results.

Run One			
Date: 20/4/20	**Time [s]**	**Regolith Collected [g]**	**Energy Use [mAh]**
Battery One	240	2155.7	113.00
Run Two			
Date: 20/4/20	**Time [s]**	**Regolith Collected [g]**	**Energy Use [mAh]**
Battery Two	240	2177	109.00

$$\frac{2155.7[g] + 2177[g]}{240[s] + 240[s]} = \frac{4332.7[g]}{480[s]} = 9.03[g/s] = \frac{(4332.7[g]/1000)}{(480[s]/3600)} \approx 32.5[kg/h] \tag{6.1}$$

$$24[V](\frac{113[mAh] + 119[mAh]}{2}) * 0.015 = 41.76[Wh] \tag{6.2}$$

Discussion and Conclusions: This test can only give an indication of regolith collecting rate. The operators ability is a major factor. Intuitive controls and ease of operation played a large part in the success of the design. We do not have our peers designs to compare against, but the results are in the mid-range of our design goal. We are confident that the results in test A could be improved by up to a factor of two given sufficient practise time.

The test can also only give an indication of the energy efficiency. As with the collection rate, a lot is dependant on the skill of the operator. The batteries used in the two tests have different internal resistance. If we ran the test again we would practise more beforehand, use the same battery for each test and and run the tests over a longer time period.

6.3.2 Test B

Aim: To determine the maximum speed.

Method: A sand and gravel driveway with a gentle slope was used to simulate a 'Mars-like surface". A five metre course was measured out and the rover timed over 10 runs up and down for a total distance of 100 metres. For each run the rover was driven as fast as possible. For the first five of these runs the rover was unloaded. For the second five it carried 1kg of regolith. The runs were timed visually with an iphone stopwatch. This means the times are only accurate to 5 tenths of a second. The rovers speed was calculated by dividing the total distance driven by the total time taken. This figure was then converted to kilometres per hour. We thought that there might be a measurable difference in speed between the loaded and unloaded runs. The loaded run was slightly faster but this could have been due to timing errors or the operator being better at negotiating the course.

Figure 6.5: The 5 metre run for test B.

Figure 6.6: The rover loaded with 1kg of regolith.

Results and Calculations:

Table 6.2: Test B. Maximum speed test results

Test One (Empty)					
Date:25/4/20	**run 1**	**run 2**	**run 3**	**run 4**	**run 5**
Time 1(s):	9.48	9.63	9.88	9.44	9.76
Time 2(s):	9.01	8.74	8.53	8.39	8.86
Run Distance (m)	5	5	5	5	5
Total distance (m)	10	10	10	10	10
Test Two (1 kg load)					
Date:25/4/20	**run 1**	**run 2**	**run 3**	**run 4**	**run 5**
Time 1(s):	9.71	9.66	10.16	10.06	9.80
Time 2(s):	8.55	8.72	8.75	8.63	8.75
Run Distance (m)	5	5	5	5	5
Total distance (m)	10	10	10	10	10

Table 6.3: Result calculations

	run 1	**run 2**	**run 3**	**run 4**	**run 5**	**Total**
Time 1[s]	9.48	9.63	9.88	9.44	9.76	
Time 2[s]	9.01	8.74	8.53	8.39	8.86	
Total [s]	18.49	18.37	18.41	17.83	18.62	*91.72*
Distance [m]	10	10	10	10	10	*50*
	run 6	**run 7**	**run 8**	**run 9**	**run 10**	**Total**
Time 1[s]	9.71	9.66	10.16	10.06	9.80	
Time 2[s]	8.55	8.72	8.75	8.63	8.75	
Total [s]	18.26	18.38	18.91	18.69	18.55	*92.79*
Distance [m]	10	10	10	10	10	*50*

$$\frac{(10[m] * 10)}{(18.26 + 18.38 + 18.91 + 18.69 + 18.55 + 18.49 + 18.37 + 18.41 + 17.83 + 18.62)[s]} = 0.54[m/s] \tag{6.3}$$

$$0.54[m/s] * (\frac{3600}{1000}) = 1.94[km/h] \tag{6.4}$$

Discussion and Conclusions:

The results give good indication of the maximum operating speed. The test was carried out on sand and small rocks. Some operator input was needed over to maintain a straight course due the the sand. This was easily achieved due to the low gear ratio and abundant torque. We had thought there might be a difference in speed with the 1kg load. However, no difference outside the expected error in the results was observed.

6.3.3 Test C

Aim: To determine the maximum operating load.

Method: A sand and gravel driveway with a gentle slope was used to simulate a 'Mars-like surface". The rover was loaded with an increasing amount of regolith and the effects observed. The load was increased until normal operation was only just achievable. The regolith was weighed and this weight recorded as the maximum operating load.

Figure 6.7: The rover during maximum load testing.

A video of the rover with maximum load is here.

Results and Conclusions: The rover was able to operate with a load of $5.8kg$. The rear gate and all controls were still operable. The tyres which are 1/10 scale RC truck tyres were compressed by the weight so the rover was unable to turn. At the maximum design load of 5 kg operation was normal, however the maximum speed by attenuated by a factor of two.

Note: The regolith for this test was not loaded via the scoop. It was poured into the container by hand. It may or may not be possible to load this amount via the scoop. The largest amount loaded via scoop was $2.5kg$.

7. Conclusion

7.1 Design Effectiveness Analysis

7.1.1 Specifications

Table 7.1: Rover Specifications

Dimensions	***Length [cm]***	***Width [cm]***	***Height[cm]***
Scoop Lowered	55	27.5	23
Scoop Raised	47	27.5	40
Weight [kg]	4.3		
Maximum Load [kg]	5.8		
Maximum Speed [km/h]	1.94		
Collection Rate [kg/h]	32.5		
Energy Use [Wh]	41.8		

7.1.2 Revisiting the Problem Statement

A successful design is a valid solution to the problem statement. Our rover is a successful design because it was completed and tested in the ten week time constraint. It is able to transit on a 'Mars-like' surface and collect resources. It has a high rate of resource collection. Each of these metrics was tested and quantified. It required very little modification when completed. It is extremely robust and is easy to manoeuvre. It is able to operate with a load more than 130 percent of its weight. During the testing phase of approximately 4 hours of continuous operation no components failed. The only maintenance required was lubrication of the actuator pivots and motor gearboxes.

The design principles we formulated and adhered to throughout the design and build process are a major factor in the success of this project. Our principal of simplicity allowed us to keep the project within the specified timeline. It meant we were able to start the build early and complete the rover with time for an extensive test program. Our design was also able to be built with limited tools and commonly available materials. This enabled us to be flexible in the design when COVID-19 denied us access to the UNSW facilities and overseas components.

7.1.3 Driving Experience

The rover was very easy to drive and operate. This is due to the use of a RC transmitter as the control interface. The control sticks are intuitive to use. with directional control on the right and scoop operation on the left. The transmitter is programmable so many controls can be customised to suit different drivers. For example the rear gate control switch can be programmed to any of the switches on the transmitter. During testing many operators (including children) were able to accurately control the rover within 1 or 2 minutes.

7.1.4 Would it work on Mars?

Our rovers suitability for Martian operation requires expert assessment that we did not have access to due to COVID-19. We consider that in most areas the rover would operate successfully on the Red Planet. It can easily transit on sand and negotiate obstacles and small rocks. It has a small lightweight power supply and does not require an oxygen atmosphere.

Some form of radiation shielding would need to be provided for the electronics. Some radiation protection is provided by most of vulnerable components being mounted under the regolith container. Assessment of the shielding required is beyond the scope of this report.

7.2 Recommendations for future improvement.

Use a better quality linear servo that is lighter and better suited to the design geometry. As mentioned above the use of a much larger linear actuator required a redesign of the scoop geometry and a large increase in the operating weight of the rover. The linear actuator pivots should also be replaced with bearings to increase the lifespan of the joints and reduce strain on the actuator.

The scoop could be redesigned to optimize regolith capacity and increase collection rate.

Use larger diameter wheels to give more ground clearance and improve the maximum speed.

The rear gate closure system could be reconfigured to the original solenoid design mentioned above. The servo implementation works well. However, at high loads the servo's are under constant load.

A fuse should be included in the power supply system as mentioned above.

The electronics could be condensed onto a single PCB to use less space and reduce the amount of wiring. Real time telemetry could be added. Real time information on current draw while operating would be useful. The current iteration only allows current measurement when physically plugged into the Arduino IDE. Some semi-autonomous features could be added such as obstacle avoidance.

7.3 Notes on the effect of COVID-19

7.3.1 Design Complexity

Due to delivery of the rotary encoder being delayed proportional scoop control was not able to be implemented. The design was simplified slightly to a binary on/off operation. We found this had little effect on the final operation.

7.3.2 Cost

The project was significantly more expensive than planned. The motor controllers, DC-DC voltage converter, Solenoid, and axles were ordered from China in February, but have still not arrived. These parts were re-brought from Australian vendors at a premium.

7.3.3 Build Quality

We had planned on using the laser cutters in the for all the plywood machining. We were only able to cut the main profiles, all other machining was completed in a home environment. The build quality is not as high as could have been achieved in a workshop.

Bibliography

[1] Liebherr 960. *Liebherr 984 loading Caterpillar 773*. URL: https://youtu.be/hakU8IVTDc4. (accessed: 11.4.2020).

[2] Duane B. *RC Channels, L293D Motor Driver - Part 2 Calibration And Code*. URL: http://rcarduino.blogspot.com/2012/05/interfacing-rc-channels-to-l293d-motor.html. (accessed: 10.4.2020).

[3] Discovery Canada. *RASSOR: NASA'S new rover that will mine the surface of Mars!* URL: https://youtu.be/sBod90pUfB0. (accessed: 11.4.2020).

[4] Largest Dam. *NASA Robotic Mining Competition - 2018 - North Dakota State University*. URL: https://youtu.be/cocg1u0nwbI. (accessed: 11.4.2020).

[5] ESA. *Mars Express*. URL: https://sci.esa.int/mars-express. (accessed: 11.4.2020).

[6] Josh Adams John-David Warren and Harold Molle. *Arduino Robotics*. New York, New York: Apress, 2011.

[7] Binghao Li. *ENGG100 Design Project: Mars Regolith Collection*. URL: %5Ctext%7Bhttps://moodle.telt.unsw.edu.au/pluginfile.php/5373592/mod_resource/content/14/ENGG1000%20MERE%20Mars%20Project%20T1%202020.pdf%7D. (accessed: 11.4.2020).

[8] Rob Manning and William L. Simon. *Mars Rover Curiosity*. Washington DC: Smithsonian Books, 2014.

[9] Scott Olberding. *NASA Robotic Mining Competition - 2018 - North Dakota State University*. URL: https://youtu.be/sBod90pUfB0. (accessed: 11.4.2020).

[10] Mobile Robots. *6WD All Terrain Robot*. URL: http://www.mobilerobots.pl/index.php?p=1_110_6WD-All-Terrain-Robot. (accessed: 11.4.2020).

[11] John Salt. *11 Things to Know About LiPo Batteries to Get the Best Performance, Life, Value Fun Out of Them, Whatever You Fly*. URL: https://www.rchelicopterfun.com/lipo-batteries.html. (accessed: 19.4.2020).

[12] Paul Scherz and Simon Monk. *Practical Electronics for Inventors*. New York, New York: McGraw-Hill Education, 2016.

[13] Jen Synclair. *A Brief History of Plywood and How It Helped Win the War*. URL: https://www.famitchell.com.au/brief-history-plywood-helped-win-war/. (accessed: 19.4.2020).

[14] Adam Steltzner with William Patrick. *The Right Kind of Crazy*. New York, New York: Penguin Random House, 2016.

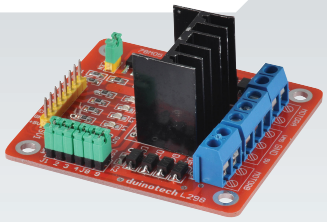
Appendix A: Datasheets

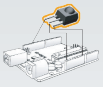
Figure 7.1: Arduino Mega

duinotech

XC4492

Dual/Stepper Motor Controller Module





- Type: Module
- Application: Add On Module
  Control the speed of two DC motors or one stepper motor
- Dimensions: 69(L) x 56(W) x 36(H)mm

Specifications	
	Dual/Stepper Motor Controller Module
Maximum Current	4A
Logic Voltage	5VDC
Operating Voltage	3VDC - 30VDC
Chipset	L29BN
Dimensions	69(L) x 56(W) x 36(H)mm
Additional Features	Status LEDs

Dual/Stepper Motor Controller Module Overview:

Using the powerful L298N Dual Motor Driver, this module allows full control of two DC Motors or one stepper-motor. Features back-EMF and over-temperature protection, it is ideally suited to drive two-motor robot kits such as our KR3130 & KR3132. An on-board 5V regulator can also be used to power your project.

What is included: 1 x Dual Motor Controller Module

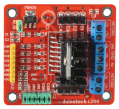
Essential Accessories: Robot Chassis (a or KR3132)

Optional Accessories: Battery Pack to provide power for motors

Did you know: There is a 5V regulator on the module which you can use to power your Duinotech mainboard, just make sure the 5V_EN jumper is in place

Pinout		
Module	Duinotech	Function
ENA	D6	Enable Motor A
IN1	D4	Motor A Forward
IN2	D7	Motor A Reverse
IN3	D3	Motor B Forward
IN4	D2	Motor B Reverse
ENB	D5	Enable A
GND	GND	Ground Connection
+5V	5V	5V
VMS	-	Power for Module (up to 30V)
GND	-	Negative Return for VMS

Dual/Stepper Motor Controller Module Sample Projects:



Jumper	Function
J9/CSB	Place a current sense resistor between these pins to measure current on channel B, or short with jumper if not using a current sense resistor.
J8/CSA	Place a current sense resistor between these pins to measure current on channel A, or short with jumper if not using a current sense resistor.
J4/UR1	Jumper on to activate 1KOhm pullup resistor on IN4 (usually not necessary if connected to Arduino).
J3/UR2	Jumper on to activate 1KOhm pullup resistor on IN3 (usually not necessary if connected to Arduino).
J2/UR3	Jumper on to activate 1KOhm pullup resistor on IN2 (usually not necessary if connected to Arduino).
J1/UR4	Jumper on to activate 1KOhm pullup resistor on IN1 (usually not necessary if connected to Arduino).
5V_EN	Jumper on to provide +5V power to the board from 5V regulator from VMS supply

Jaycar Electronics

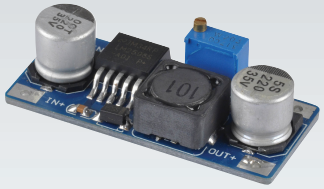
Distributed by:
TechBrands by Electus Distribution Pty. Ltd.
Ph: 1300 738 555
www.techbrands.com
Made in China

Figure 7.2: L298N Motor Controller

duinotech

XC-4514

DC- DC Stepdown Module



- Type: Module
- Application: Add On Module
  Step Down DC voltages.
- Dimensions: 49(L) x 26(W) x 12(H)mm

DC - DC Stepdown Module Overview:

This module accepts any voltage from 4.5 - 35VDC, and outputs any lower voltage from 3-34V. Output is adjusted via a multi-turn potentiometer. Use it to run your 5V Duinotech projects from a 6v, 9v or even 12v Supply

Note: With new modules, the pot might be at its highest setting. You will have to turn the potentiometer around 10 times counter clockwise before you see the voltage change.

Specifications

	DC - DC Stepdown Module
Input Voltage	4.5V - 35V
Output Voltage	3V - 34V
Maximum Current	2.5A
Dimensions	49(L) x 26(W) x 12(H)

What is included: 1 x DC - DC Stepdown Module

Essential Accessories: 8xAAA Holder (PH9209)

Optional Accessories:

DC - DC Stepdown Module Sample Projects:

Did you know:
The dropout voltage can be as low as 0.8V when the unit is supplying 1A, and it more efficient than the voltage regulator built into the Duinotech UNO Board

Jaycar Electronics

Distributed by:
TechBrands by Electus Distribution Pty. Ltd.
Ph: 1300 738 555
www.techbrands.com
Made in China

Figure 7.3: DC DC Converter

Appendix B: Arduino Code

Test Code

```
// connect motor controller pins to Arduino digital pins
// Linear Actuator - A

int LIN_enB = 46;
int LIN_in3 = 50;
int LIN_in4 = 48;

// Left front - Motor - A
int LF_enA = 2;
int LF_in1 = 3;
int LF_in2 = 4;

// Left Back- Motor B
int LB_in3 = 5;
int LB_in4 = 6;
int LB_enB = 7;

// Right Front- Motor B
int RF_enB = 8;
int RF_in4 = 9;
int RF_in3 = 10;

// Right Back - Motor A
int RB_in2 = 11;
int RB_in1 = 12;
int RB_enA = 13;

void setup()
{
  // set all the motor control pins to outputs
  pinMode(LIN_enB, OUTPUT);
  pinMode(LIN_in3, OUTPUT);
  pinMode(LIN_in4, OUTPUT);

  pinMode(RB_enA, OUTPUT);
  pinMode(RB_in1, OUTPUT);
  pinMode(RB_in2, OUTPUT);

  pinMode(RF_in3, OUTPUT);
  pinMode(RF_in4, OUTPUT);
  pinMode(RF_enB, OUTPUT);

  pinMode(LF_enA, OUTPUT);
  pinMode(LF_in1, OUTPUT);
  pinMode(LF_in2, OUTPUT);

  pinMode(LB_in3, OUTPUT);
```

```
  pinMode(LB_in4, OUTPUT);
  pinMode(LB_enB, OUTPUT);
}

void demoOne()
{
  // this function will run the motors in both directions at a fixed speed
  // turn on Linear Actuator for 6 seconds
  digitalWrite(LIN_in3, LOW);
  digitalWrite(LIN_in4, HIGH);
  // set speed to 200 out of possible range 0~255
  analogWrite(LIN_enB, 255);

  delay(6000);
// Reverse linear actuator for 6 seconds
  digitalWrite(LIN_in3, HIGH);
  digitalWrite(LIN_in4, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(LIN_enB, 255);

  delay(6000);

//
//// Set Motors to forward for 6 seconds
////Right Back
//  digitalWrite(RB_in1, LOW);
//  digitalWrite(RB_in2, HIGH);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(RB_enA, 255);
//
////Right Front
//  digitalWrite(RF_in3, HIGH);
//  digitalWrite(RF_in4, LOW);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(RF_enB, 255);
//
////Left Front
//  digitalWrite(LF_in1, LOW);
//  digitalWrite(LF_in2, HIGH);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(LF_enA, 255);
//
////Left Back
//  digitalWrite(LB_in3, HIGH);
//  digitalWrite(LB_in4, LOW);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(LB_enB, 255);
//
//  delay(6000);
//
// // Set Motors backwards for 6 seconds
////Right Back
//  digitalWrite(RB_in1, HIGH);
//  digitalWrite(RB_in2, LOW);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(RB_enA, 255);
//
////Right Front
//  digitalWrite(RF_in3, LOW);
//  digitalWrite(RF_in4, HIGH);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(RF_enB, 255);
```

```
//
////Left Front
//  digitalWrite(LF_in1, HIGH);
//  digitalWrite(LF_in2, LOW);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(LF_enA, 255);
//
////Left Back
//  digitalWrite(LB_in3, LOW);
//  digitalWrite(LB_in4, HIGH);
//  // set speed to 200 out of possible range 0~255
//  analogWrite(LB_enB, 255);
//
//  delay(6000);

}

void loop()
{
  demoOne();
  delay(1000);
}
```

Operating Code

```
/**
*This RC rover operating code is based on the ExplorerBot source code in Chapter 8
*of Arduino Robotics by Warren, JD et al.
*Used with permsision.
*
*****************************************************************************
*Title: ExplorerBot_mixedSteer
*Author: Warren, JD
*Date: 1-8-2010
*Code version: 1.0
*Availability: https://sites.google.com/site/arduinorobotics/home/chapter8
*****************************************************************************
*
*Controls 3x L298N motor controllers connected to 5x 24 Volt motors for RC operation of
*a Mars Regolith Collector prototype.
*The 4 drive motor channels have full 0-100% high-resolution pwm speed control
*The Scoop channel is bi-directional ON/OFF.

*****************************************************************************
*Title: Rover_mixedSteer_v3
*Author: Sharp, D
*Date: 15-4-2020
*Code version: 3.0
*****************************************************************************
*
*/

// Inputs from RC receiver
int ppm1 = 47;
int ppm2 = 49;
int ppm3 = 51;

// connect motor controller pins to Arduino digital pins
// Linear Actuator - Motor B

int LIN_enB = 46;
int LIN_in3 = 50;
int LIN_in4 = 48;

// LEFT
// Left Front - Motor A
int LF_enA = 3; // PWM
int LF_in1 = 2;
int LF_in2 = 4;

// Left Back - Motor B
int LB_in3 = 5;
int LB_in4 = 6;
int LB_enB = 9; // PWM

// RIGHT
// Right Front - Motor B
int RB_enB = 10; //PWM
int RB_in4 = 7;
int RB_in3 = 8;

// Right Back - Motor A
int RF_in2 = 12;
int RF_in1 = 13;
```

```
int RF_enA = 11; //PWM

unsigned int servo1_val; // unsigned integer simply means that it cannot be negative.
int adj_val1;
int servo1_Ready;

unsigned int servo2_val;
int adj_val2;
int servo2_Ready;

unsigned int servo3_val;
int adj_val3;
int servo3_Ready;

unsigned int servo4_val;
int adj_val4;
int servo4_Ready;

////////////////////////////////

int deadband = 100; // sets the total deadband - this number is divided by 2 to get the deadband for each direction. Higher value will yield larger neutral band.
int deadband_high = deadband / 2; // sets deadband_high to be half of deadband (ie. 10/2 = 5)
int deadband_low = deadband_high * -1; // sets deadband_low to be negative half of deadband (ie. 5 * -1 = -5)

// You can adjust these values to calibrate the code to your specific radio - check the Serial Monitor to see your values.
int low1 = 900;
int high1 = 2000;
int low2 = 900;
int high2 = 2000;

int x; // this will represent the x coordinate
int y; // this will represent the y coordinate

int left; // X and y will be converted into left and right values
int right; // X and y will be converted into left and right values

int speed_low;
int speed_high;

int speed_limit = 255;

int speed_max = 255;
int speed_min = 0;


void setup() {

  TCCR1B = TCCR1B & 0b11111000 | 0x01; // change PWM frequency on pins 9 and 10 to 32kHz
  TCCR2B = TCCR2B & 0b11111000 | 0x01; // change PWM frequency on pins 3 and 11 to 32kHz

  Serial.begin(9600);

  // set all the motor control pins to outputs
  pinMode(LIN_enB, OUTPUT);
  pinMode(LIN_in3, OUTPUT);
  pinMode(LIN_in4, OUTPUT);

  pinMode(RF_enA, OUTPUT);
  pinMode(RF_in1, OUTPUT);
```

```
  pinMode(RF_in2, OUTPUT);

  pinMode(RB_in3, OUTPUT);
  pinMode(RB_in4, OUTPUT);
  pinMode(RB_enB, OUTPUT);

  pinMode(LF_enA, OUTPUT);
  pinMode(LF_in1, OUTPUT);
  pinMode(LF_in2, OUTPUT);

  pinMode(LB_in3, OUTPUT);
  pinMode(LB_in4, OUTPUT);
  pinMode(LB_enB, OUTPUT);

  //PPM inputs from RC receiver
  pinMode(ppm1, INPUT);
  pinMode(ppm2, INPUT);
  pinMode(ppm3, INPUT);

  delay(1000);

}


void pulse(){

  servo1_val = pulseIn(ppm1, HIGH, 20000); // read pulse from channel 1
  // make sure servo 1 value is within range (between 800 and 2200 microseconds)
  if (servo1_val > 800 && servo1_val < 2200){
    servo1_Ready = true;
  }
  else {
    servo1_Ready = false;
    servo1_val = 1500;
  }

  servo2_val = pulseIn(ppm2, HIGH, 20000); // read pulse from channel 2
  if (servo2_val > 800 && servo2_val < 2200){
    servo2_Ready = true;
  }
  else {
    servo2_Ready = false;
    servo2_val = 1500;
  }

    servo3_val = pulseIn(ppm3, HIGH, 20000); // read pulse from channel 3
  if (servo3_val > 1800){
    raise_scoop();
  }

  else if (servo3_val < 1200){
    lower_scoop();
  }
  else{
    stop_scoop();
  }
}



void loop() {

```

```
  pulse(); // read pulses

  ////////////////////////////////

  if (servo1_Ready) {

    servo1_Ready = false;
    // map servo value from 1500 microseconds (neutral) to a value of 0.
    //If pulse is above neutral (forward) value will be 0 to 255, otherwise it will be 0 to
    -255 for reverse
    adj_val1 = map(servo1_val, low1, high1, -speed_limit, speed_limit);
    adj_val1 = constrain(adj_val1, -speed_limit, speed_limit);

    x = adj_val1;

  }
  if (servo2_Ready) {

    servo2_Ready = false;

    adj_val2 = map(servo2_val ,low2, high2, -speed_limit, speed_limit);
    adj_val2 = constrain(adj_val2, -speed_limit, speed_limit);

    y = adj_val2;

  }



  if (x > deadband_high) {  // if the Up/Down R/C input is above the upper threshold, go
    FORWARD

    // now check to see if left/right input from R/C is to the left, to the right, or centered
    .

    if (y > deadband_high) { // go forward while turning right proportional to the R/C left/
    right input
      left = x;
      right = x - y;
      test(); // make sure signal stays within range of the Arduino capable values
      left_forward(left);
      right_forward(right);
      // quadrant 1
    }

    else if (y < deadband_low) {   // go forward while turning left proportional to the R/C
    left/right input
      left = x - (y * -1);  // remember that in this case, y will be a negative number
      right = x;
      test();
      left_forward(left);
      right_forward(right);
      // quadrant 2
    }

    else {   // left/right stick is centered, go straight forward
      left = x;
      right = x;
      test();
      left_forward(left);
      right_forward(right);
    }
```

```
238   }
239
240   else if (x < deadband_low) {    // otherwise, if the Up/Down R/C input is below lower
        threshold, go BACKWARD
241
242     // remember that x is below deadband_low, it will always be a negative number, we need to
        multiply it by -1 to make it positive.
243     // now check to see if left/right input from R/C is to the left, to the right, or centered
        .
244
245     if (y > deadband_high) { // // go backward while turning right proportional to the R/C
        left/right input
246       left = (x * -1);
247       right = (x * -1) - y;
248       test();
249       left_reverse(left);
250       right_reverse(right);
251       // quadrant 4
252     }
253
254     else if (y < deadband_low) {   // go backward while turning left proportional to the R/C
        left/right input
255       left = (x * -1) - (y * -1);
256       right = x * -1;
257       test();
258       left_reverse(left);
259       right_reverse(right);
260       // quadrant 3
261     }
262
263     else {   // left/right stick is centered, go straight backwards
264       left = x * -1;
265       right = x * -1;
266       test();
267       left_reverse(left);
268       right_reverse(right);
269     }
270
271   }
272
273   else {     // if neither of the above 2 conditions is met, the Up/Down R/C input is centered
        (neutral)
274
275     if (y > deadband_high) { //spin right
276
277       left = y;
278       right = y;
279       test();
280       left_forward(left);
281       right_reverse(right);
282
283     }
284
285     else if (y < deadband_low) { //spin left
286
287       left = (y * -1);
288       right = (y * -1);
289       test();
290       left_reverse(left);
291       right_forward(right);
292
293     }
```

```
294
295     else {
296
297       left = 0;
298       right = 0;
299       left_stop();
300       right_stop();
301     }
302
303   }
304
305   Serial.print(left);
306   Serial.print("   ");
307   Serial.print(right);
308   Serial.print("    ");
309
310   Serial.print(servo1_val);
311   Serial.print("   ");
312   Serial.print(servo2_val);
313   Serial.print("    ");
314   Serial.print(servo3_val);
315   Serial.println("   ");
316 }
317
318
319 int test() {
320
321   // make sure we don't try to write any invalid PWM values to the h-bridge, ie. above 255 or
      below 0.
322
323   if (left > 254) {
324     left = 255;
325   }
326   if (left < 1) {
327     left = 0;
328   }
329   if (right > 254) {
330     right = 255;
331   }
332   if (right < 1) {
333     right = 0;
334   }
335
336 }
337
338
339 // Create single instances for each motor direction, so we don't accidentally write a shoot-
      through condition to the H-bridge.
340
341 void right_forward(int m2_speed){
342   analogWrite(RF_enA, m2_speed);
343   digitalWrite(RF_in1, LOW);
344   digitalWrite(RF_in2, HIGH);
345   digitalWrite(RB_in3, HIGH);
346   digitalWrite(RB_in4, LOW);
347   analogWrite(RB_enB, m2_speed);
348 }
349
350 void right_reverse(int m2_speed){
351   analogWrite(RF_enA, m2_speed);
352   digitalWrite(RF_in1, HIGH);
353   digitalWrite(RF_in2, LOW);
```

```
  digitalWrite(RB_in3, LOW);
  digitalWrite(RB_in4, HIGH);
  analogWrite(RB_enB, m2_speed);
}

void left_forward(int m1_speed){
  analogWrite(LF_enA, m1_speed);
  digitalWrite(LF_in1, LOW);
  digitalWrite(LF_in2, HIGH);
  digitalWrite(LB_in3, HIGH);
  digitalWrite(LB_in4, LOW);
  analogWrite(LB_enB, m1_speed);

}

void left_reverse(int m1_speed){
  analogWrite(LF_enA, m1_speed);
  digitalWrite(LF_in1, HIGH);
  digitalWrite(LF_in2, LOW);
  digitalWrite(LB_in3, LOW);
  digitalWrite(LB_in4, HIGH);
  analogWrite(LB_enB, m1_speed);
}

void right_stop(){
  digitalWrite(RF_in1, LOW);
  digitalWrite(RF_in2, LOW);
  digitalWrite(RB_in3, LOW);
  digitalWrite(RB_in4, LOW);
}

void left_stop(){
  digitalWrite(LF_in1, LOW);
  digitalWrite(LF_in2, LOW);
  digitalWrite(LB_in3, LOW);
  digitalWrite(LB_in4, LOW);
}

void raise_scoop(){
  digitalWrite(LIN_in3, HIGH);
  digitalWrite(LIN_in4, LOW);
  analogWrite(LIN_enB, 255);
}

void lower_scoop(){
  digitalWrite(LIN_in3, LOW);
  digitalWrite(LIN_in4, HIGH);
  analogWrite(LIN_enB, 255);
}

void stop_scoop(){
  digitalWrite(LIN_in3, LOW);
  digitalWrite(LIN_in4, LOW);
  analogWrite(LIN_enB,0);
}
```

Appendix C: Purchasing List

Component List			
No	**Description**	**Quantity**	**Vendor**
Main Components			
1	Interior Plywood Sheet 897mm*600mm*7mm	1	Bunnings
2	Interior Plywood Sheet 1200mm*810mm*3mm	1	Bunnings
3	Threaded Stainless Steel Rod 1200mm*6mm	1	Bunnings
4	Angle Bracket 25mm*25mm*40mm	3	Bunnings
5	M5*35mm Machine Bolt	3	Bunnings
6	M5 Flat Washer 35Pk	1	Bunnings
7	M5 Hexagon Nut 20Pk	1	Bunnings
8	M6 Nylon Lock Nut 6pk	2	Bunnings
9	M6 Nylon Stainless Steel Nut 6pk	2	Bunnings
10	M3*15 mm Machine Bolt	3	Bunnings
11	M3 Flat Washer 35Pk	1	Bunnings
12	M3 Hexagon Nut 200Pk	1	Jaycar
13	Threaded Stainless Steel Rod 1200mm*8mm	1	Bunnings
14	Corflute	1	Bunnings
15	Rustoleum Spray Paint (Orange & Gray)	2	Bunnings
16	MEGA 2560 R3 Arduino ATmega2560	1	eBay
17	37mm 24V DC High Torque Gear Box Motor	4	eBay
18	Module DC-DC Volt Regulator Arduino Comp	1	Jaycar
19	DC24V 100mm Multi-function Linear Actuator	1	eBay
20	Spektrum AR6210 2.4G 6CH Receiver	1	Secondhand
21	Arduino Compatible Stepper Motor Controller Module	3	Jaycar
22	Arduino Compatible 5A Current Sensor Module	1	Jaycar
23	15 Gauge Hookup wire (Red & Black)	2	Jaycar
24	Mini Brass Spade Connector 100Pk	1	Jaycar

Appendix D: YouTube Playlist

A YouTube playlist of the build and testing videos is here.