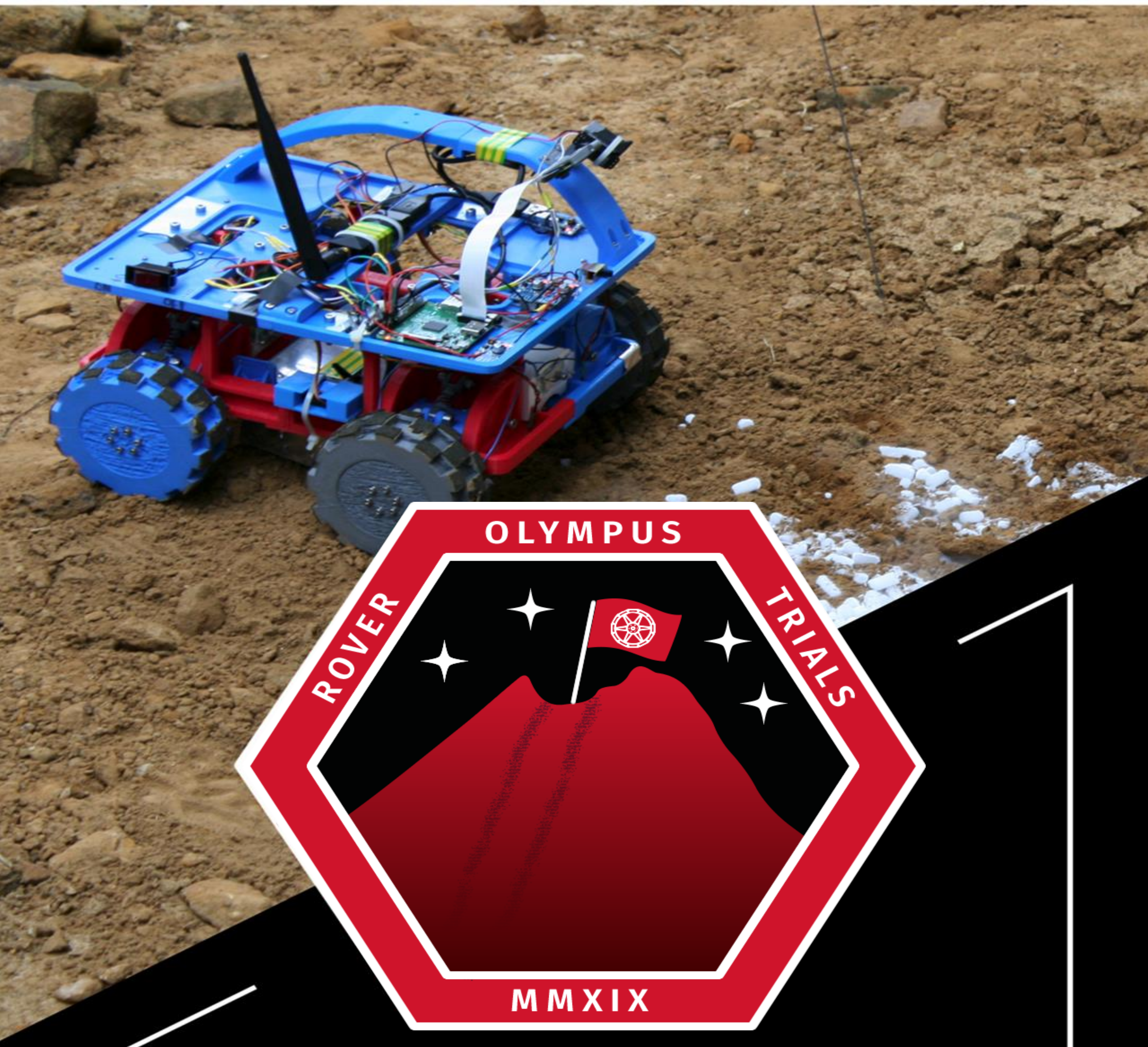


OLYMPUS ROVER TRIALS

MMXIX

Ares

Kingston University London

2019 - 2020

INTRODUCTORY STATEMENT

From the PDR to the CDR, a few people have left the team as they are no longer interested, and our roles have changed significantly, such as a change in project manager and removing titles such as secretary. Due to people leaving the team unannounced and a relatively low amount of team members contribution, we are not at the stage we expected or wanted to be at. We intend to compensate for this during our build and test phase of the rover as we hav

CONTENTS

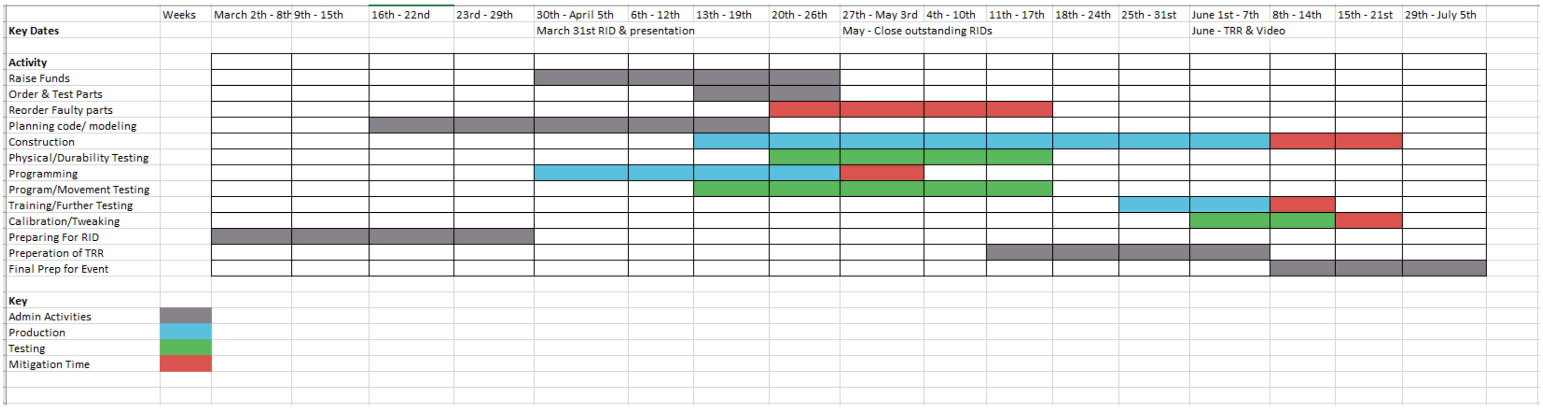
1.0 PROJECT MANAGEMENT

1.1 Team Roles & Roster

Name	Role	Year/Course	Contact Details
Bertie Kemp	Project Manager, Lead Engineer	Year 1 Aerospace Engineering with Astronautics & Space Technology	K1910784@kingston.ac.uk
Emily Scothern	Lead Programmer, Lead Electronics	Year 1 Computer Science	K1940601@kingston.ac.uk
Iras Charles	Retired Project Manager, Researcher	Year 1 Aerospace Engineering with Astronautics & Space Technology	K1819154@kingston.ac.uk
Nicolo Fasciano	Researcher	Year 1 Aerospace Engineering with Astronautics & Space Technology	K1942892@kingston.ac.uk

1.2 Schedule

Once we have passed the CDR review and had the meeting with our reviewer, we are going to start our schedule, such as launching our funding plan and our development of building the rover. We are waiting until we go through the CDR review process to make detailed solid works models and start prototyping as once we have a final design that both we and our reviewer is happy with, we can start making models to manufacture the rover from. During the time when we are waiting for this review, we will still be able to progress as we already posses personally some components that we intend to use, such as motors and Arduinos, and we can possibly get some materials from our University. This allows us to test and proceed whilst waiting for the review and feedback.



We will try and stick to this Gantt chart in order to stay on task, however we may need to increase the length of some tasks if we run into delays.

1.3 Budget and Pricing

We have included a rough price list of all the components we will need. Once we have passed the CDR review and have a final design, we will finalise the price list and bill of materials to make sure we are not wasting money and that we are being sensible with our spending.

Part	Price	Amount Needed	Total	Link	Purpose & Justification
Raspberry Pi 4 (4GB model B)	£54.00	1	£54.00	https://	The Raspberry Pi 4 is the latest and most power
Arduino Nano 33 BLE Module	£21.06	1	£21.06	https://	The small form factor and 9-axis intertial senso
Small breadboards (option 1)	£5.99	2	£11.98	https://	Space is very limited in this project and we do r
Small breadboards (option 2)	£5.15	2	£10.30	https://	Thin version from RS components, for advantag
Jumper wires	£5.95	1	£5.95	https://	Good variety of different wires to use for attatc
Battery	£49.99	1	£49.99	https://www.amazon.co.uk/FCONEGY-11-1V-5500mah-	
Motors	£7.99	4	£31.96	https://www.amazon.co.uk/EsportsMJJ-Teeth-Electric-	
Stepper Motors	£11.58	1	£11.58	https://www.amazon.co.uk/Kuman-Stepper-arduino-2	
Tracks	£11.13	1	£11.13	https://www.amazon.co.uk/Baosity-Plastic-Removable	
Suspension systems			£0.00		
Cameras	£24.00	2	£48.00	https://	Only raspberry pi camera module suitable for li
ultrasonic sensors	£8.99	3	£26.97	https://www.amazon.co.uk/ELEGOO-Ultrasonic-Raspbe	
Claw	£13.39	1	£13.39	https://www.aliexpress.com/item/32860356446.html?s	
Total	£219.22		£296.31		

Our total price comes to £296.31, however there are a few things missing from this list that we will be able to find and include when we need to raise funds. There are some materials and components that we may be able to get from our University without having to pay, or with a discount, this may reduce our final total. We are planning on raising £250 in order to be matched by UKSEDS grant for a total of £500, we don't expect to need all £500 and will not spend anything over what we need. In order for us to raise £250, we will be using our University's society backing scheme, KUBacker. If we are able to raise £125 through donations, we will be matched by the University, giving us £250.

1.4 Requirements Update

Due to further reading of the guidelines, research and exploring a deeper insight of what we would like our solution to be, our requirements have changed and expanded drastically. These requirements shall also include those listed in the guidelines as these are vital to the project, as well as some extra requirements specific to our design.

Base requirements are the requirements the rover must achieve by the end of the project, and ones we fully expect to achieve, whereas the extra requirements are added goals we would like the rover to be capable of, but do not necessarily expect to achieve.

Base Requirements:

The rover must/shall…

- Weigh no more than 5kg.
- Occupy no more than $0.03m^3$.
- Have sufficient interface to attach to the vibration plate.
- Resist up to at least 4.5 GRMS of random vibration.
- Remain statically stable in all directions to an angle of at least 30 degrees.
- Be capable of movement over a surface of sand and rocks up to 15 cm.
- Be capable of being statically stable at 30 degrees without slipping, falling, or turning over.
- Not spin its tracks on soft materials such as sand and mud.
- Be able to operate under sub-optimal conditions such as rain.
- Be capable of travelling at least 40m from the operator without losing connection.
- Be capable of easily manoeuvring around larger rocks that it cannot traverse over.
- Always have a stable connection to the operating laptop.
- Be able to be manually moved from the operating laptop.
- Feed a live surrounding view to the connecting laptop.
- Have as little potential points of failure as possible, e.g. All wiring connections must be stable, all code robust, all physical structures durable and well-constructed etc.
- Have the battery last for at least 30 minutes of usual running. Battery must not be lead acid and must be safe.
- Include a kill switch to automatically switch off all rover electronics/ halt the rover.
- Pose limited hazards to itself and those surrounding it.
- Accurately pick up the sample, preferably without dropping it.
- Safely transport a sample back towards the operator.

Extra Requirements:

The rover should…

- Be capable of autonomous movement towards the location of the sample.
- Autonomously recognise and pick up a sample.
- Reroute itself around obstacles it cannot traverse.
- Have a battery life that can extend to an hour of running.
- Be durable and rugged, able to withstand bad weather conditions and falling from a height of 1m.
- Be capable of moving over rocks of 25cm in height.
- Be capable of moving over a 50-degree incline.
- Have a strong connection at least 50m from the operator
- Have a sliding cover to seal the bottom of the rover from the elements.

All requirements previously indicated in our PDR have been theoretically addressed in our design of the rover, however we cannot physically address any of these requirements until prototyping/ construction of our rover has begun (see introductory statement). We plan to make up for this shortcoming and loss of time through rigorous designing and planning which should help us meet our requirements quickly and with little need to build additional prototypes or acquiring new hardware.

1.5 Trade-Offs

There are not many trade-offs we have had to make so far in designing our rover, but we do anticipate having to make a few trade-offs as we progress with the project. Here are a few we suspect we may have to make:

Having up to 13 ultrasonic sensors would provide a full 360-degree coverage of obstacle detection, however we may encounter bandwidth and coding limitations preventing this from being possible.

Having two high-capacity batteries may not be possible due to price limitations.

Having two high quality streaming cameras and many computationally intensive automation scripts may be too much for the Raspberry Pi to handle.

We have determined that we are thankfully unlikely to encounter trade-off issues when it comes to weight due to our calculated estimate of the weight of the current design is roughly 4kg. 4kg is a worst-case scenario, and we expect the actual weight to be lower. We are going to design as if it is 4kg as this allows our design to still be valid if we are wrong.

1.6 Project Risks & Management

We have several risks that could delay our progress. The main risk is if we receive faulty or broken parts, this obviously will slow down our manufacturing, especially if the part was essential to the rover. We plan to mitigate this, if it happens, by working on other tasks that we do have the parts for until we can get a replacement. If we have a part that takes a long time to deliver, this is also a risk as some parts we plan to order may take up to a month to be delivered, and if they arrive broken , we may not be able to replace them in time.

Another risk is if we find out during construction that a design won't work and that we will have to change it in order to progress. We will mitigate this by being very thorough with our designs, making sure we think of every way it could not work, and showing it to others to see if they can spot anything, we as a team, have missed.

Another risk is our design not meeting our requirements. In order to mitigate this risk, we will try and catch this risk as early as possible, we plan to discuss each week of manufacturing if we have any issues such as our rover not meeting requirements.

Overall mitigation plans also include having weekly progress meetings to ensure that we are always on time, and to divide up tasks and work out how to solve any issues that come up to tackle it, and inclusion of mitigation time into the schedule in case delivery of parts or certain tasks take too long. This means that even if delays or issues occur, we should be able to deliver the final rover on time.

Our final risk is if we run into issues when coding and programming our rover. We plan to mitigate this by making sure we have all the required software and a good plan, with flowcharts and pseudo code.

1.7 Testing Plan

To test our design, we will start by constructing our aluminium extrude frame and by 3D printing our wheel modules (explained in 2.1.2). We will then get the track and see if our motors are powerful enough to move our rover with weights to simulate the final weight of the rover, we are planning to use a bench power supply for this test.

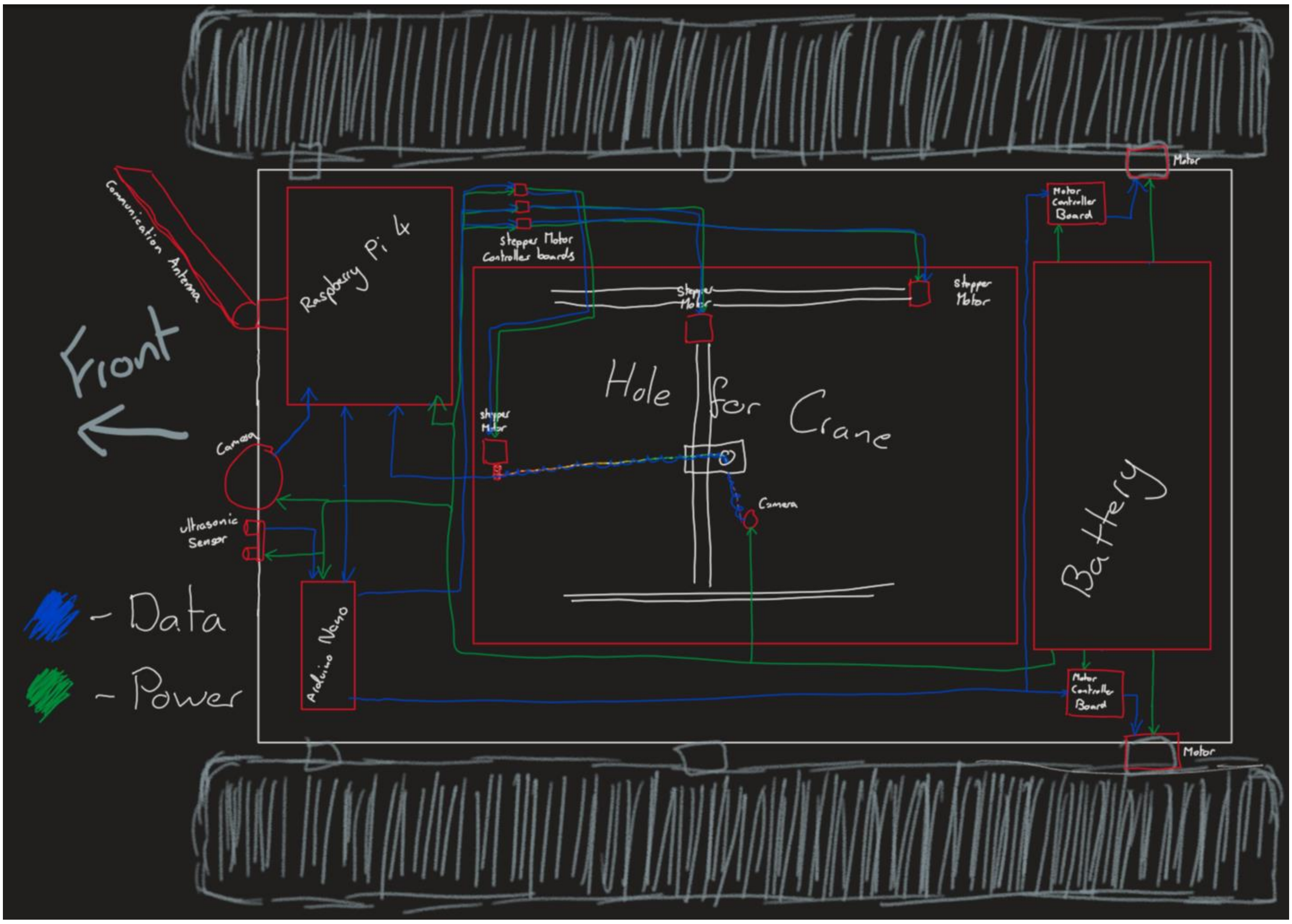
Once we have confirmed that our design for movement works, we are going to perform two tests in tandem, the crane system for picking up a sample, and the electronics for connecting from the laptop to the rover. We will test this by making sure we are able to freely move the crane around the opening in the bottom of the rover, and if we encounter any issues with the wire or inaccuracies, we may have to quickly design a different solution, however we are confident in this design and don't expect to have to redesign it. The electronics will be the easiest to test as we can complete this test without funding and at any time as within our team, we personally almost all the components.

We will then test our connection. It is important to us to have a strong, reliable connection, so we intend to connect a laptop to the rover, then try and interfere with that connection and improve our code until we are sure that the rover can communicate up to 50m away, and with poor conditions such as poor weather or interference.

2.0 Full Rover Design

2.1 Mechanical Subsystems

Rover Overview:



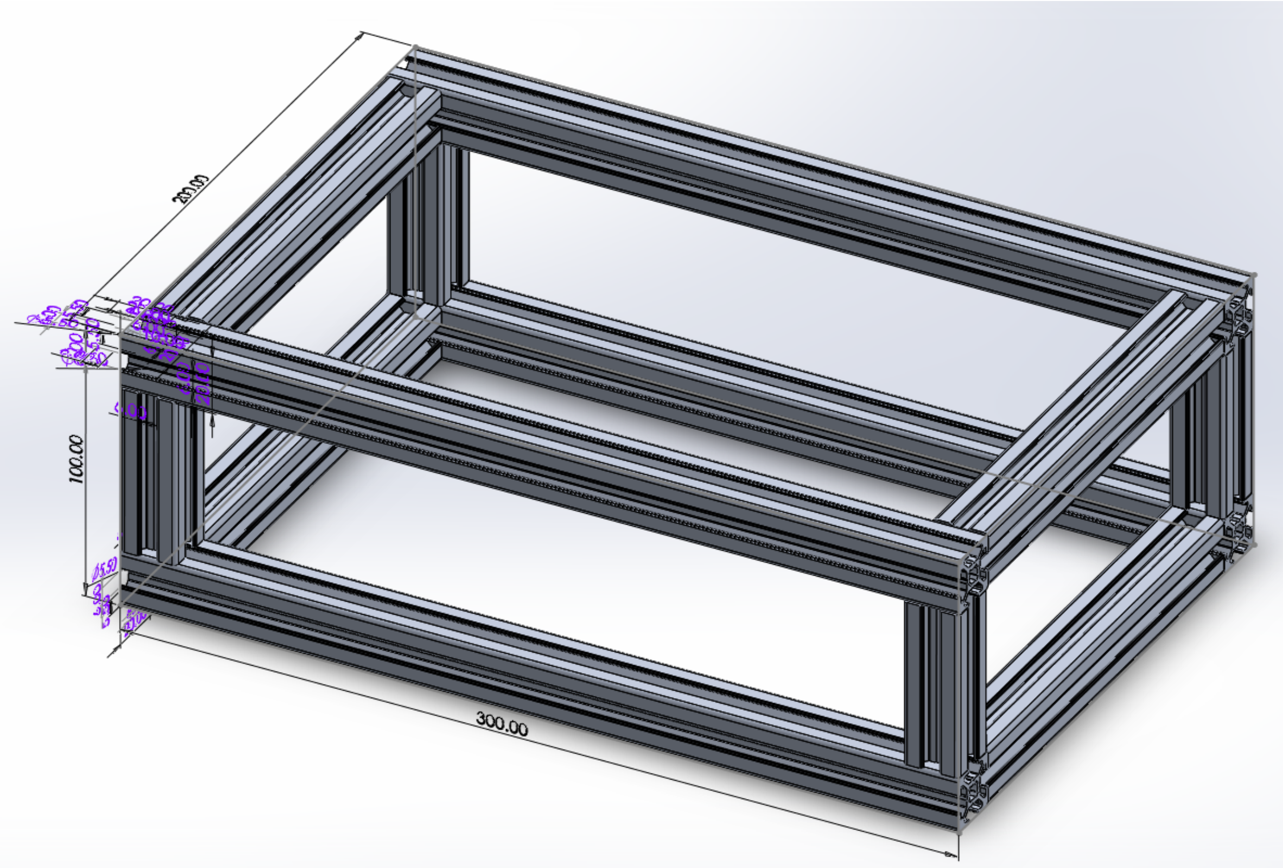
2.1.1 Chassis

The chassis will be made from aluminium extrusion, sometimes called 'Rexroth'. We will be using 4 sections 300mm long, and 4 sections 160mm. The sections will be 20mm width and height. This will just be a frame, with components such as the wheels and the electrical components bolted on to it. This allows us to bolt components on anywhere and move them around, for example, when we tension the caterpillar tracks. To fill in the gaps in the frame on all sides, we will be using wood panelling during the prototype stage, allowing us to easily drill holes and/or change our configurations. When we have finalised our design we plan on changing this to a stronger material such as a sheet of aluminium, or even some carbon fibre panels which we may be able to salvage from an existing rover at our university, although we are not confirming the use of carbon fibre. Having these side panels also allows us to be protected from poor weather and fulfils our basic requirement 'Be capable to operate under sub-optimal conditions such as rain'.

Due to our frame being able to be screwed/bolted into at all points, we plan to be able to connect to the vibration interface plate by bolting through are frame, and into the plate. We plan to use 3 bolts on each edge to do this. This fulfils our basic requirement 'Have sufficient interface too attach to the vibration plate'.

Our chassis is 0.3x0.2x0.1m, giving us a total volume of 0.006m^3. This fulfils our basic requirement 'Occupy no more than 0.03m^3'.

The dimensions are shown in our solid works model:

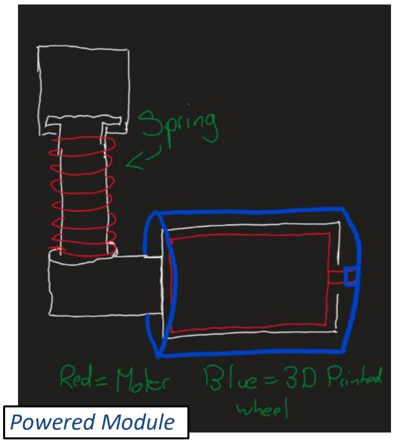


We only made a solid works model of the chassis as this is the one design that we are extremely confident that this will not change due to any concerns of the reviewer.

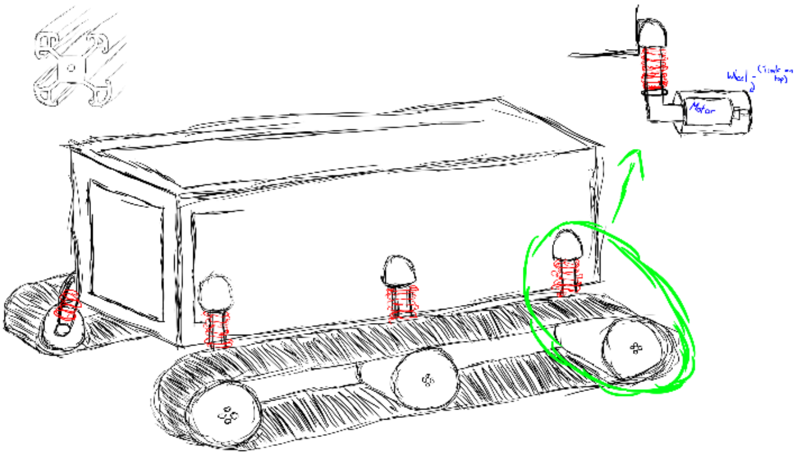
2.1.2 Tracks

We are using caterpillar tracks rather than a traditional method such as 4 normal wheels or even a rocker-bogie system as we believe it will give us an advantage over these other systems. Caterpillar tracks are better than standard wheels as there isn't as much pressure on the rover due to the greater surface area provided by the tracks. Caterpillar tracks also don't run into the issue of having a wheel stuck in the air as in all but the rarest scenarios, some part of the track will always be in contact with the ground. Caterpillar tracks have the advantage over a rocker-bogie system as being slightly less complex, as with a rocker-bogie, you need to design and implement a differential and make sure each section can support the stress of the rover. There are some downsides to using a caterpillar track however, such as it being slightly complex to get the tension of the belt correct, as if this is wrong it can result in less traction, poor grip or even the system tearing itself apart, causing the track to come off the wheels.

The tracks will be supported and connected to the rover by 3 points (modules). At each 3 points they can either be powered or unpowered.



Powered Module



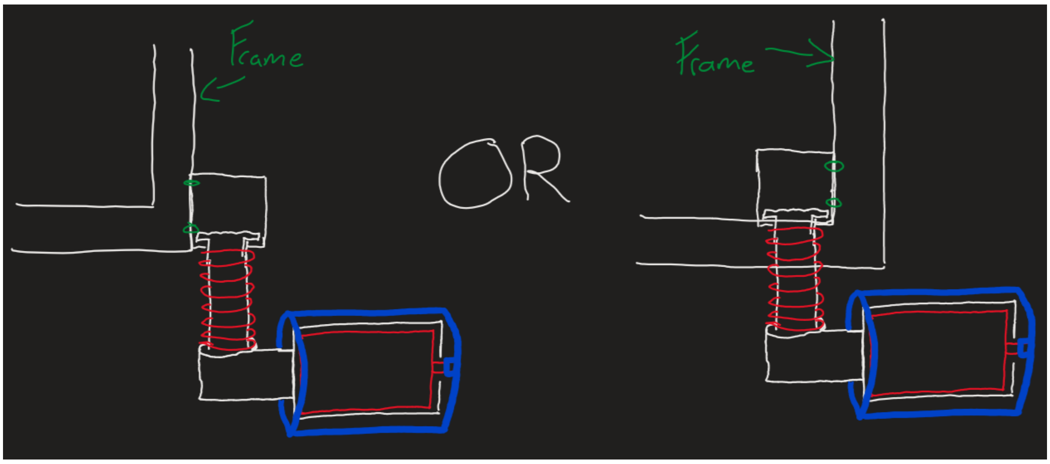
As we have not yet made a prototype, we will be testing out what configuration will be best for the motors. Our initial plan is to have one motor at the back, whilst the other modules contain just a bearing connected to the wheels, allowing the treads to be tensioned whilst also allowing it to move freely, being moved by the motor. These modules will be bolted in two places (to stop them twisting, possibly resulting in the tension reducing) to the aluminium frame. The aluminium extrusion allows us to do this at any point along the frame, which we will use to tension the caterpillar track when installing it. We will first bolt on the centre module, then place the track around it, followed by placing the other two modules within the track. We will then pull each outer module to the ends of the rover, until the tracks are at the correct tension. They will be bolted into place, keeping the tension on the track.

Testing will allow us to find the best placement of these motors. We will be using a maximum of 4 motors, as if testing shows that we could need a motor in each end of the tread, then that's the maximum number of motors we will need. We will start by building all 6 'modules', as we are going to make it easy to swap out the motor with a bearing. We will buy 2 motors to start off with, as we do not want to have extra unnecessary parts which we do not use. The motors will mount directly to the 3D printed wheels, which will then turn the tracks.

In the event that we are unable to get a working rover with the tracks, we still have the ability to take the tracks off, then 3D print some wheels to place on each module. If this were to happen, we would need 4 motors, but we would still be able to complete our design.

We are undecided of how to attach the modules to the chassis, either from the outside or from within the inside of the rover. We will wait until we have built and started prototyping to finalise this decision.

The tracks giving us traction and more grip than traditional wheels allows us to fulfil our basic requirement 'Remain statically stable in all directions to an angle of at least 30 degrees'.



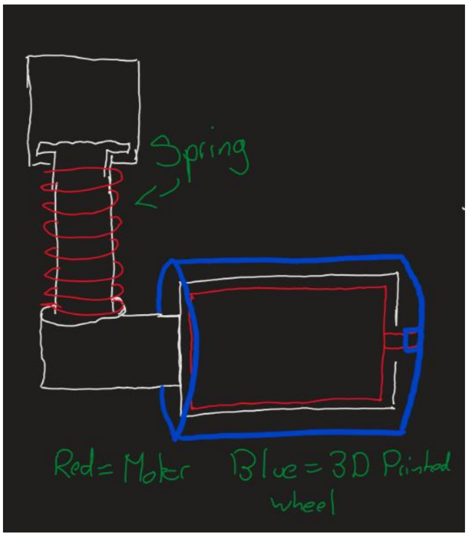
The tracks will have suspension, connect to each module, which we will explain more in the next section.

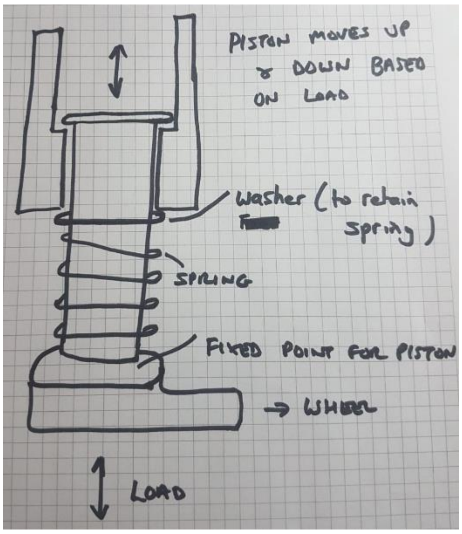
2.1.3 Suspension

In order to reduce vibrations to the electronics and other components as much as possible, we will have suspension on all 6 of our modules. This allows any small bumps from quickly driving over a rock to be dampened and absorbed by the spring. We have done some research into a system to provide suspension and what would be a suitable choice to buy in, however we have an idea on how to construct our own version and will be using the buy in choice as a backup if we fail to make an adequate suspension. This buy in option will require some adapter plates to mount if we fail to make our custom version, however this is not a huge concern as these can be quickly 3D printed and tested.

We plan to make a suspension system by having a 'piston' inside a cylinder, trapped by a lip on the cylinder, with plenty of room above for the piston to move up into. The piston will extend out the bottom of the cylinder and have another lip at the bottom of it. A spring will be placed between the bottom lip of the piston and the bottom face of the cylinder. This will extend and push the piston into the lip of the cylinder, and then when some force is applied to the bottom of the piston, the spring will absorb some of that force and compress, allowing the piston to move up in the cylinder. This will allow the sudden force to be absorbed by compressing the spring and moving the piston, rather than be transferred to the chassis of the rover.

The piston and cylinder will first be 3D printed to ensure our concept works. We will then test the 3D printed version to see if it can withstand the forces and stresses we intend to put it through. If it does not survive this, then we will weigh up the pros and cons of continuing our design and making it out of a stronger material, such as aluminium, or if it's better to just buy in a system that we have already researched.





[Nicolo] Research into buy in option:

In order to guarantee the best performances and to avoid any damage to the internal parts or to a hypothetic load, it's important to design a very good suspension system for the rover. The rover will have 6 wheels and every one of those will be directly connected to the chassis through a shock absorber. The distance between the shock absorbers will be fixed on one end of the body, whereas, on the other it will be adjustable so that we can set the tracks at the right tension. As we don't know what the final weight of the whole system will be, we will be using adjustable shock absorbers. That will allow us to modify the length of the springs and consequently the thrust produced following Hooke's law: F= -kX

Where k is the spring constant which remains the same, and X is the displacement of the spring. As we can see the force produced is proportional to the extension, so, if we modify the last one, we can get the right force for our system (Williams, M, 2015).

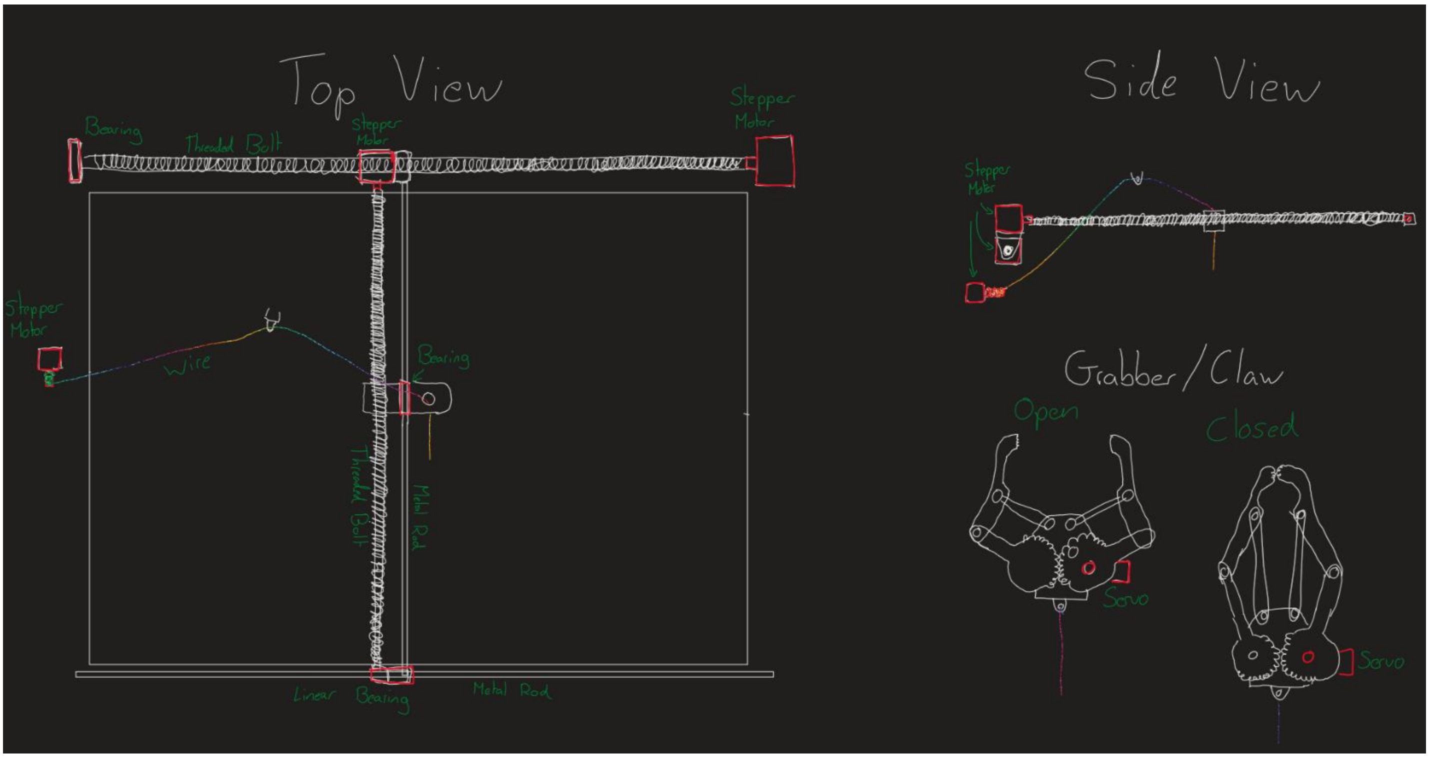
In conclusion these are the shock absorbers, we need 3 sets of them (6 in total):

2.1.4 Crane System

We decided to use a crane system instead of a traditional system like a robotic arm or a 'scoop' as it helped to keep our rover contained within the frame of the chassis. This also allows us to have finer control over how we retrieve the sample, as instead of moving the entire rover, we have the precision of the stepper motors to move our claw precisely over the sample. In addition to this, we noticed from research into previous rovers undertaking the Rover Olympus Trials, that robotic arms usually come loose during the vibration test and that scoops placed on the underside of the rover can sometimes cause the rover to 'run aground' and fall over. By using a crane system mounted internally, it will not get caught on rocks and will not break off during the vibration test.

For the crane, we will be using stepper motors and threaded bolts, similar to how 3D printers' function on their x y and z axis. We will be using this system for the x and y of our crane system, then use a stepper motor and a wire to lower and retract our claw, consisting of a servo which uses gears to open and close the claw arms. Stepper motors allow us to be very precise as they are repeatable and are very good at controlling speed and precision. Their low speed torque also contributed to their precision. This is important for our crane system as we want to be able to place our claw directly over the sample and collect it as efficiently as possible, requiring as little attempts to grab as possible.

Diagram to show visually how our crane system will function:



2.2 Electronic Subsystems

The electronics of the rover will consist of:

- Raspberry Pi 4 (4GB model B) – This will handle the camera feeds, network connection to the operator device, and any autonomous scripts.
- Arduino Nano 33 BLE – This will handle the movement of all motors and monitor sensor data, passing any relevant data onto the Raspberry Pi.
- Breadboards and jumper wires – To connect sensors and motors to the Arduino.
- Battery – One battery will be used to supply power to the entire electronic system (excluding external operator connections).
- 2 Small high-torque geared motors – These will be used to drive the track, ultimately moving the rover.
- 3 Stepper motors – These will make up the crane system used to pick up and retrieve the sample.
- 1 Servo Motor – This will be used to control the grabber used in the crane sample retrieval system.
- 2 V2 Raspberry Pi camera modules – These will be used to both stream image to the operator and analysed to conduct autonomy implemented into the final rover. They will be located at the front of the rover, and in the middle of the crane system to provide 2 vital views.
- 1 – 13 Ultrasonic distance sensors – These will detect the distance to nearby surfaces and warn the operator/ feed into the autonomy system to help avoid collisions.
- 2 Motor controller boards – These will serve as an interface between the Arduino and the two different sorts of motors. A simple motor controller board can be used for the driving motors, and a stepper driver module can be used for the stepper motors.
- 2 Switch mode regulator boards – These will be used to reduce the voltage passed through them from the power supply so that the Raspberry Pi and Arduino can receive the correct voltage power.
- Raspberry Pi multi camera adaptor module – This is needed to connect two of the camera modules to the Raspberry Pi.
- 2 Raspberry Pi camera module flex extension cables – These are needed to extend the connection length so that we can position the cameras in optimal positions.
- Micro USB Cable – To connect the Arduino to the Raspberry Pi.
- Micro SD Card – To hold the operating system of the Raspberry Pi on.
- A laptop running custom software will be used to control the rover over a WIFI or ethernet connection.
- 2 USB Wi-Fi antennas – These will allow both the rover and the laptop to maintain a strong connection.

The current 'Architecture' Diagram of the electronic systems inside the rover is displayed below. This purely represents how the components will connect to each other and is not representative of the final design as the electronics will be built to fit around the mechanical systems of the rover and is therefore highly likely to change.

2.2.1 Arduino

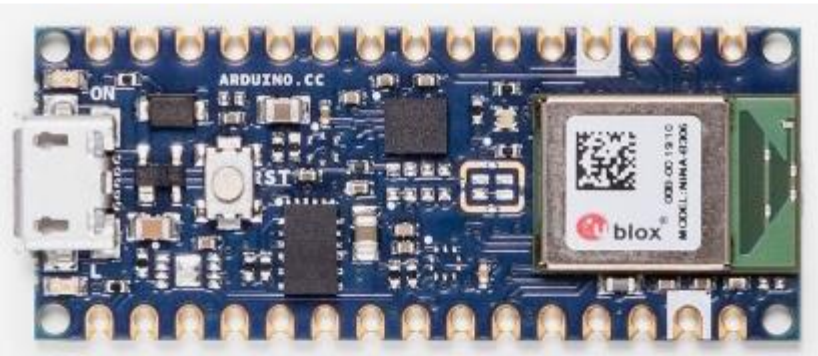
We will be using the Arduino Nano 33 BLE to drive the motors and monitor the ultrasonic sensor modules. It will be connected to the Raspberry Pi 4 through USB-C and act as a slave board to it. The motors and sensors will connect to the Arduino nano through wired connections. The Arduino will be vital in order to move the motors and therefore is vital to the functioning of the rover and contributes to a majority of the requirements.

We decided to use an Arduino for controlling the motors and the sensors as Arduinos provide an easy way to connect and control parts of the system by providing easy pin connectors and the Arduino IDE. This IDE allows us to easily code at a base level, assigning voltages to specific pins using the built-in libraries and then export code to the board. Using an Arduino for the rover will make the programming of the motor control far easier, quicker and more convenient.

The Arduino Nano 33 BLE, in particular, was selected due to many factors:

1. Small Form-Factor – The Arduino Nano is only 45x18 mm, which is useful for our rover due to limited space. Using this small form-factor board in conjunction with small breadboards to increase the pinout quantity allows us flexibility to fit the electronics around the mechanical systems.
2. Micro-B Port – The inclusion of a micro-USB port is useful as it allows the Arduino to easily communicate with the Raspberry Pi.
3. Cheap Price – The low price of this controller allows us to save money to use elsewhere in the rover whilst not having to make any compromises on functionality of the board.
4. 9 Axis Inertial Measurement Unit – This capability is the most useful of the Nano board range in our specific application, as this data may be useful towards helping the rover move safely whilst autonomous.

Layout:



(Arduino, 2020)

Key Specifications:

Microcontroller	nRF52840
Operating Voltage	3.3V
CPU	32-bit ARM® Cortex™-M4 CPU @64 MHz
CPU Flash Memory	1MB (nRF52840)
Digital Input / Output Pins	14
SRAM	256KB (nRF52840)
USB	Micro-USB
Length	45 mm
Width	18 mm
Weight	5 gr (with headers)
Extra Features	Bluetooth Host & Client Capability 9 Axis Inertial Measurement Unit (Incl. accelerometer, gyroscope & magnetometer)

(Arduino, 2020)

2.2.2 Raspberry Pi

We will be using the Raspberry Pi 4 Model B with 4GB of RAM to manage the camera feed and handle the autonomous scripts of the rover. It will send the camera feeds to the operator device and receive user instructions from this device via a strong Wi-Fi connection. It will also have the Arduino connected to one of the USB-3 ports and over this connection it will receive sensor data and send motor instructions either forwarded from the user or generated by the autonomous scripts running on the Raspberry Pi. The use of the Raspberry Pi 4 is not necessarily vital to the core movement of the rover but is important in fulfilling many of the core and extra requirements including: 'Feed a live surrounding view to the connecting laptop', 'be able to be manually moved from the operating laptop', 'be capable of autonomous movement towards the location of the sample', and 'autonomously recognise and pick up a sample'.

We decided to use this primarily due to the powerful CPU and large quantity of RAM. This is due to our desire for this main computer of the rover to stream two high-quality camera feeds and run scripts including machine-learning solutions that will require a high amount of computing power and multiple processing threads. The Raspberry Pi 4 is one of the only microcomputers on the market capable of this, and other solutions we have considered, such as the BeagleBone, are more expensive and less well-documented, making the Raspberry Pi 4 our best option. It also has integrated Wi-Fi capabilities, which we intend to use as our main mode of communication, with an ethernet interface we can use as a backup communication, should our WIFI connection fail for any reason. Finally, it also has a lightweight operating system designed specifically to work with it and its easy-to-add modules that we can use to program in high-level languages, further adding to the advantages of using it as the computer for our rover.

Using the Raspberry Pi 4 does come with the trade-off of being a rather large and bulky component with a high power-draw that will struggle to find a place in our chassis, but we deemed the convenience and capability worth having to work around this fact.

Layout:

(The PiHut, 2020) 'Raspberry Pi 4 Model B'

Key Specifications:

CPU	64-bit ARM Cortex-A72 CPU (Quad-Core @ 1.5GHz)
Operating Voltage	5V
Memory	4GB LPDDR4-3200
Interfaces	Raspberry Pi standard 40 pin GPIO header CSI (camera) DSI (display)
Connectivity	Gigabit Ethernet

	Dual-band Wi-Fi Bluetooth 5
Ports	2x Micro-HDMI 2x USB 3 2x USB 2 USB Type-C Power Port MicroSD slot
Dimensions	88 x 58 x 19.5 mm
Weight	46g

(Piltch, A, Halfacree, G.,2019), (Sneddon, J.,2020)

2.2.3 Sensors

Within the rover there will be two different types of sensors:

(1) Two V2 Raspberry Pi camera modules with 8megapixel sensors will be fitted onto the rover. They both are capable of streaming 1080p video at 30Hz and will be used both to feed a video stream to the operator controlling/monitoring the rover, and feed into object-detecting or machine learning algorithms as part of the rover's autonomy. These cameras will be positioned with one on the front of the rover to direct general movement, and one above the crane assembly to monitor the operation of the crane system and to direct the retrieval of the sample. This subsystem partially fulfils the requirement 'Feed a live surrounding view to the connecting laptop' by being capable of recording and streaming live, high-quality video, as well as being vital to the extra requirement of autonomous movement and sample retrieval.

The modules will connect to the Raspberry Pi through a multiple camera adaptor module as it only has one CSI port and we will be using two. We may also require longer flex ribbon cables for these modules to comfortably sit in the correct place and route around the other subsystems inside the rover.

We chose these specific cameras as they provide enough resolution at a high enough frame rate to allow an operator to control the rover's movement based solely on the camera feed, and for image detection scripts to be conducted on the camera feed. Another advantage to using these specific camera modules is that they use the specific CSI ribbon connector that is present on the Raspberry Pi 4 and are heavily supported within the Raspbian OS we will be using. This will make them much easier than other 3rd party options to connect and program, making them worth the slightly higher price.

We may have to compromise on the second camera found on the bottom of the rover to a lower spec model to reduce the load on the processor as streaming two 1080p cameras may prove very intensive for the CPU, which could slow it down. This camera makes a suitable compromise as less detail is needed in the bottom image due to the sample being closer to the camera and easier to detect.

Layout:

(The PiHut, 2020) Raspberry Pi Camera Module V2

Key Specifications:

Still resolution	8-megapixel, 3280 x 2464
Video Modes	1080p30, 720p60 and 640x480p90
Connection Type	Raspberry Pi CSI Ribbon Connector
Raspbian Support	Yes
Dimensions	25mm x 23mm x 9mm
Weight	< 4g

(The PiHut, 2020) Raspberry Pi Camera Module V2

(2) Ultrasonic Sensors will be used to detect the distance between the rover and the ground or potential obstacles. We plan to have at least two on the rover, one on the front of the rover to detect oncoming obstacles, and another on the bottom of the rover to detect the distance of the base plate from the ground to avoid the rover becoming stranded. These values will feed into an algorithm to determine whether the rover must move around an obstacle if it cannot travel over it to fulfil the requirement 'Be capable of easily manoeuvring around larger rocks that it cannot traverse over'.
The sensors will be connected to the Arduino through jumper cables and a breadboard, which may later be swapped out for soldered wiring for durability.

To decide on which sensor to use for this task, we compared ultrasonic, infrared and photoelectric sensors and how they would perform in the context of our rover. Firstly, an infrared sensor works by emitting a pulse of infrared light and detect whether or not the signal is returned, meaning that it can only detect whether or not an obstacle is present, and not the distance (Miller, L., 2019). This means that despite it being the cheapest of these options, it is unsuitable for our use-case as we need to detect the distance from the object so that the operator or autonomous script can determine how to avoid the obstacles surrounding it in time. The photoelectric sensor also uses a pulse of light and can detect the distance of objects accurately at long distances but can become unreliable when introduced to artificial light or sunlight that can interfere with it. They are also far too expensive to be a viable option for our budget (Telemecanique Sensors., N/A). As our testing will take place outside during the day, this sensor is also unsuitable. Finally, the ultrasonic sensor is the best distance sensor for our application as they are low-cost, common modules that use sound to estimate the distance of an object; meaning that it works regardless of the colour or lighting conditions of the object it is sensing. The weak point of ultrasonic sensors is a short range and ineffectiveness against soft, foam-like materials that absorb sound waves (Keller Technology, 2018). As the rover will not be encountering any foam-like materials or require a long range for our specific application, so luckily none of these shortcomings of ultrasonic sensors will be disadvantageous to us.

In the future we may find that additional sensors would be useful to further prevent the rover from becoming stranded or stuck and wish to try more sensors in other positions; but this will not be a problem as these sensors are often sold cheaply and in packs of 3 or more.

2.2.4 Motor Systems

We will be using brushed 12v motors in order to have enough torque to move the caterpillar tracks and to be able to climb obstacles without lacking the power to get over them. We will need a basic speed controller for these motors; however, these will be much cheaper and simpler than if we had gone with brushless motors. We are not planning on needing very long-term reliability and as we will make sure the brushed motors are driven correctly, we are not worried about burning the motors out, however we will keep a close eye on this during testing and development. We chose these motors as they are relatively cheap as we may need 4 of them, and they will give us more than enough performance. The motors are around 200g so we would prefer to use as little as possible, preferably 2, however if we need to use 4 of them, we can afford 1kg of weight to be used for motors.

2.2.5 Battery, Powering & Wired Connections

In order to power our rover, we will need a few different voltages. We will need 12v to drive our motors, 5v to drive our raspberry pi and a few other components such as stepper motors and servos, and 3v to power the Arduino and any other low voltage components. The easiest way to do this is to have a single large battery pack which we can use to power everything. As lead acid batteries are not allowed, we are going to use a lithium polymer battery as its energy density is high. In order to get around 12v, we will need a 4-cell battery, which provides around 11.1v.

As no one in our team is too familiar with electronics, we have reached out to friends and family of which are electrical engineers and RC Hobbyists. With these connections we feel confident we can come up with a system that can adequately power our rover.

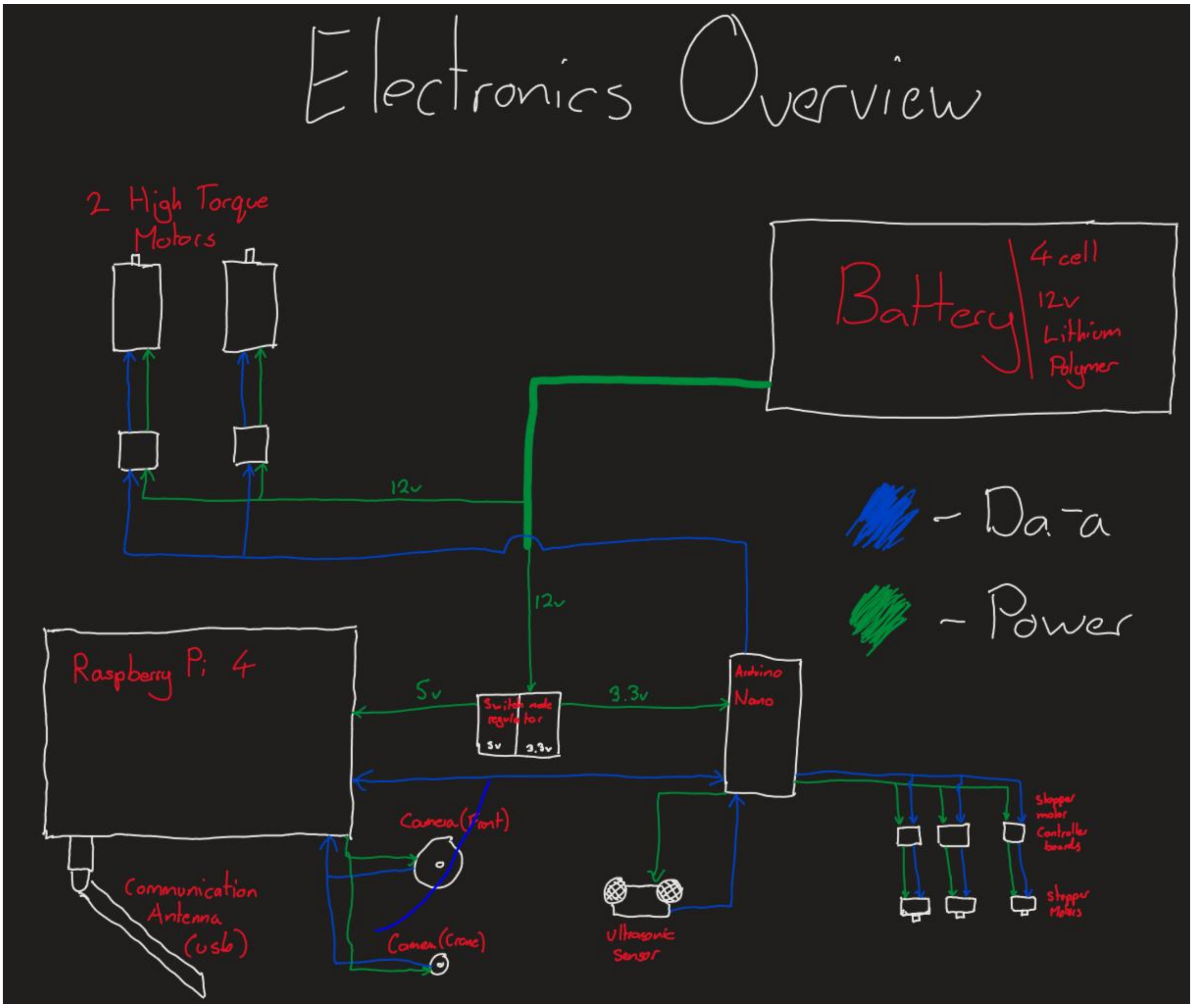
As the battery will be outputting around 12v, we will need to step that voltage down in order to get our other electronics working correctly at the right voltage. To do this we will be using switch mode regulator boards. The outputs to which will be at 5v and 3v (we will be using 2 as we need 3 different voltages, 12v, 5v and 3v).

In order to make sure our rover can operate for the entire duration of the competition (30 mins), we need to have a suitable sized battery. We have been advised that 5,000mAh should be enough, however, to add a safety margin to this, we want to have around 7,000mAh. We know that this may not be required and will add weight, so if during testing we find this to be more than enough, we may make some changes and return a higher capacity battery for a lower capacity one that is lighter. We have also been advised that we might need a 6-cell 22.2v battery in order to maintain 12v for

30 mins, however we are unsure and would need to do some tests to work out what our best option would be.

In order to make sure we don't have any short circuits or electrical issues, we will make sure we use the correct components for the job, such as thick wire for high current wires and making sure all the components are receiving their correct input voltage.

The way we are wiring our electronic components together, both power and data, is shown in this diagram:



2.2.6 Network Connection

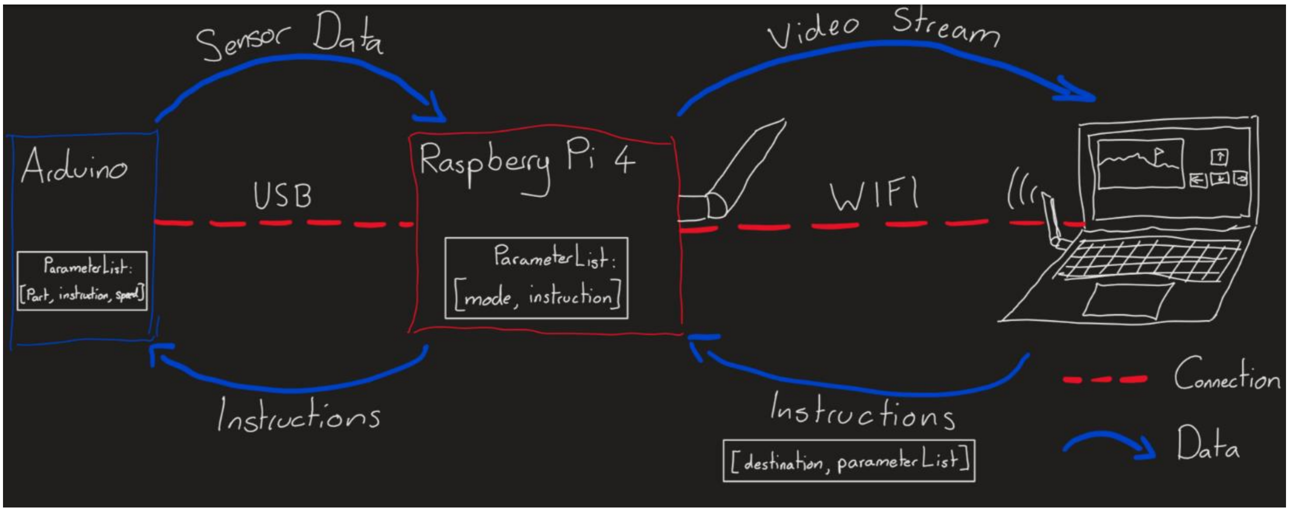
To communicate with the rover, we have opted to use a laptop running custom-made software with a WIFI connection to the Raspberry Pi 4 on board the rover. Both the Laptop and the Raspberry Pi 4 will have in-built WIFI connection capabilities, but in order to maintain a strong connection between the two, we will equip both with USB-3 WIFI antennas to strengthen the connection between these two. Over this strong connection, the Raspberry Pi will send video streams to the laptop, and the laptop will send instructions from the user back. This primary method of communication will be thoroughly tested and reinforced as it is vital to fulfilling the requirements: 'Always have a stable connection to the operating laptop', 'be able to be manually moved from the operating laptop', 'feed a live surrounding view to the connecting laptop'.

However, If the primary communication of WIFI happens to fail, there will be a secondary ethernet connection designed into the software, that can use the full ethernet port onboard the Raspberry Pi 4 to provide this connectivity. As a third, emergency mitigation technique, we will use the Bluetooth header onboard the Raspberry Pi 4 to connect to a Bluetooth game controller. We will have to code this into the software and bring a games controller with us on the day of testing, however this third backup communication system should ensure that we can still participate even if our first two communication systems manage to fail.

When choosing our primary communication method, we had many methods to decide from, primarily WIFI, Bluetooth, and Radio. Bluetooth is unsuitable as the main form of communication because whilst being easy and cheap to implement, the range is very poor and would not meet even our base requirements reliably. As for Radio, this method is very reliable with a very long range, however transmitting complex or large quantities of data over this method becomes complex and requires a lot of extra expensive equipment. WIFI is easy to implement into code and can handle large volumes of data quickly, with most devices having WIFI capability integrated into them and most buildings having wireless access points to use. This is the most viable option due to this ease of implementation and low cost. The range also fulfils the requirement of operating at least 40m away from the operator.

2.3 Software Subsystems

The main three platforms: Raspberry Pi, Arduino Nano, and Operator Laptop will each have their own software running which will communicate with each other as demonstrated in the diagram below.



On the right is the Operator laptop. This will be running custom software providing the user interface to monitor and control the rover. This interface will include two ‘modes’ the operator can switch between when controlling the rover, one for general movement and one for operating the crane. When in either mode, the corresponding camera feed will be streamed to the software in a large viewing window. Each mode will contain a different set of commands that can be used to control the system, depending on which system they are currently controlling. It will also include options to perform automatic procedures such as ‘Scan for Flag’. This program will have a connection to the Raspberry Pi 4 (In the middle) via Wi-Fi over which it will receive the video feed and send the instructions.

The instructions sent from the operator device to the Raspberry Pi 4 will be an array in the format [destination, parameterList] where destination is a string and parameterList is a list including all the information the destination processor requires. Once received by the Raspberry Pi, if the destination of the instruction is for the Raspberry Pi, it will complete the process using its own software, but if it is for the Arduino, it will be forwarded to the Arduino Nano (On the left) for it to process via a USB connection between the Raspberry Pi and the Arduino.

The Parameter List for the Raspberry Pi are as simple as reporting the current mode of the operator software (whether it is in general or crane control mode) and giving the instruction as a string. From there, the high-level software running on the Raspberry Pi will work out what to do.

If the destination of the instruction is for the Arduino, or if the Raspberry Pi automation scripts generate an instruction for the Arduino, it will have a parameter list of [part, instruction, speed]. The Arduino needs more direct guidance due to having less powerful programming, so this instruction set gives it all the information it needs. The parameter ‘part’ is a string indicating the ID of the motor/s that needs to move, the ‘instruction’ will be a Boolean telling which way the motor needs to spin, and finally ‘speed’ which will provide how fast it needs to move. The Arduino will also constantly be monitoring the ultrasonic distance sensors, and if anything significant is detected, an alert will be sent to the Raspberry Pi via the USB connection.

2.3.1 Arduino Code

The code loaded onto the Arduino will be written in C and loaded onto the board. This script will start running when the Arduino receives power and continue until it loses power or is otherwise told to halt. Therefore, we will build in a kill switch to halt the script and therefore all movement immediately as a safety feature. This also fulfils one of our core requirements.

As the Arduino is responsible for moving the motors and monitoring the sensors, the script will consist of two threads: one constantly checking the sensors, and one awaiting commands. The thread checking the sensors will generate an alert to send to the Raspberry Pi to warn it/ the operator of nearby objects. The other thread awaiting commands will read incoming instructions and execute whichever of the following functions it is told to complete, with whatever parameters it is given:

- Move Forward at [Speed]
- Move Backward at [Speed]
- Turn Left
- Turn Right
- Step Crane Forwards
- Step Crane Backwards
- Step Crane Right
- Step Crane Left
- Step Crane Down
- Step Crane Up
- Open Grabber
- Close Grabber

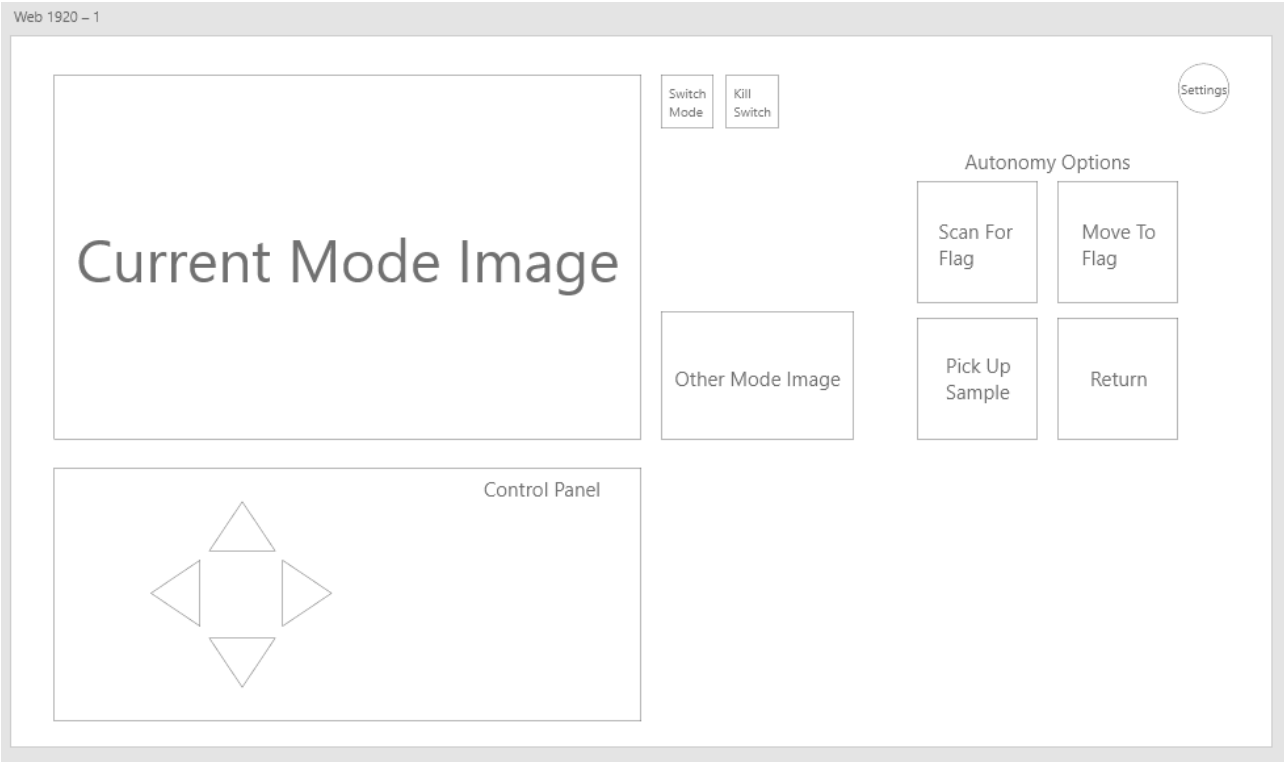
2.3.2 Raspberry Pi Code

The Raspberry Pi 4 will be loaded with a copy of the Raspbian OS on a Micro SD card. On this OS any high-level programming language scripts can be run. The raspberry Pi will be responsible for the network connection to the laptop, forwarding instructions to the Arduino and running automation scripts. The task of receiving and sending information will be performed by one program written in C# running continuously. The automation scripts will likely be written in Python and will include a combination of colour isolation, image detection and machine learning scripts to provide automation tasks. The exact functioning of these programs are yet to be planned and worked out, but time for this is set aside in the schedule and mitigation time is also added to the schedule in case of delays with programming these difficult scripts.

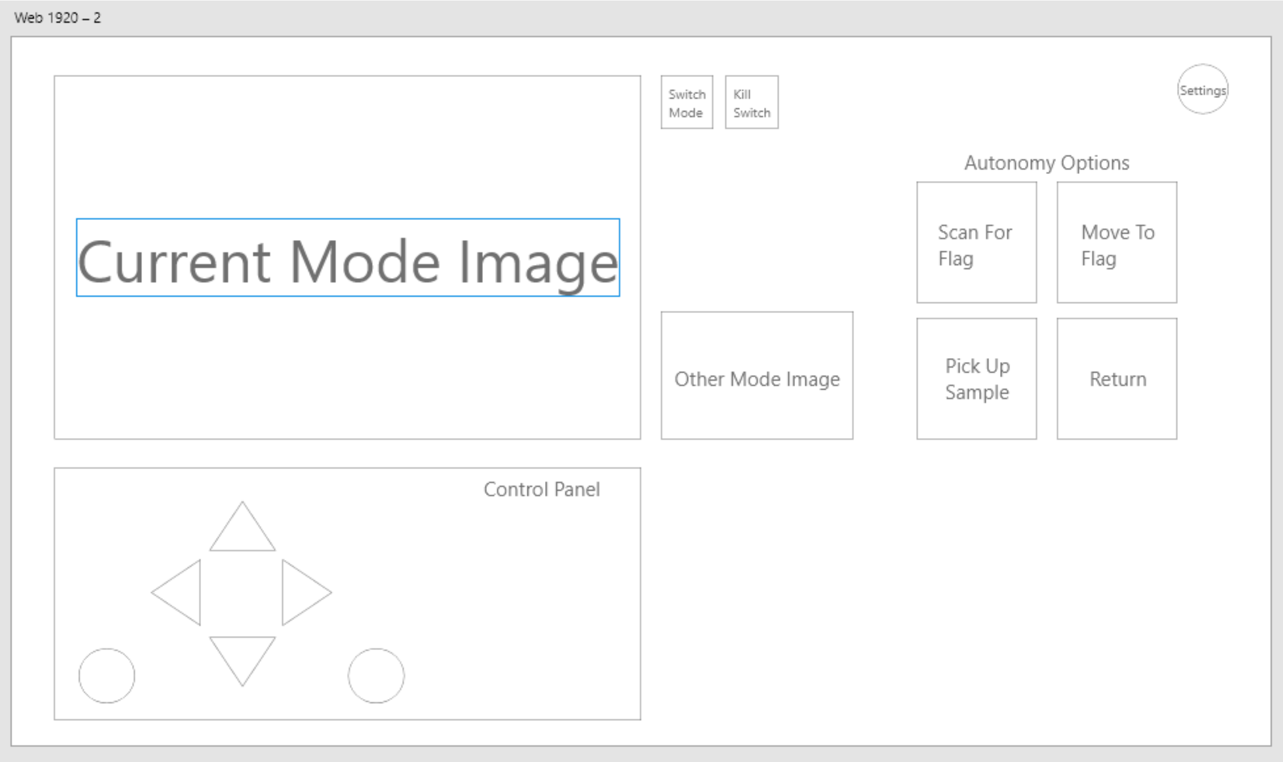
2.3.3 Laptop Interface

The custom software running on the operator laptop, as mentioned prior, will have two main modes and therefore two main interfaces. Below is a first concept image of what this interface would look like to demonstrate the sort of control options that the operator will have over the rover. This is a first wireframe image and therefore is not representative of what the final interface or even what the final functionality of the software will look like.

'General' Mode:



'Crane' Mode:



This software will satisfy the requirements of providing a live video feed to the operator and allowing the rover to be remotely controlled.

When any button in this interface is pressed, an instruction is generated by the software and sent to the Raspberry Pi 4 over Wi-Fi using the format previously detailed, depending on where the instruction needs to be sent and what the instruction will cause to happen. Any button pressed on the control panel will generate an instruction with the destination of 'Arduino'.

3.0 Safety

In order to keep our rover safe, we are going to keep the lithium polymer battery in a bag designed to protect against any malfunctions, we are going to use a balance charger and make sure we monitor the battery whilst charging. When at the competition day we will make sure the battery is quickly accessible and easy to unplug, also equipping it with a fuse and a kill switch on the outside of the rover. We will make sure that no wires are exposed, and that no exposed point of our rover is able to shock anyone or carry a voltage higher than 12v. The kill switch will allow anyone to cut power to the rover immediately.

The kill switch also allows our rover to be safe when it comes to autonomy, as if it goes off by itself or is uncontrollable, a flick of the switch will kill power and prevent it from injuring someone or damaging property. In order to prevent this, we will also make sure our code is robust so that it is not possible for the rover to go off by itself and be out of control. The code will also include a virtual kill switch that halts all script occurring within the Arduino, meaning all movement should halt unless there is some issue with the code.

4.0 Citations

Williams, M (2015) *'What is Hooke's Law?'*. Available at: https://phys.org/news/2015-02-law.html (Accessed: 12 February 2020).

Arduino, (2020) *'ARDUINO NANO 33 BLE'*. Available at: https://store.arduino.cc/arduino-nano-33-ble (Accessed: 01/03/2020)

The PiHut, (2020) *'Raspberry Pi 4 Model B'.* Available at: https://thepihut.com/products/raspberry-pi-4-model-b?variant=20064052674622&src=raspberrypi (Accessed: 01/03/2020)

Piltch, A, Halfacree, G (2019) *'Raspberry Pi 4 Review: The Gold Standard for Single-Board Computing'.* Available at: https://www.tomshardware.com/uk/reviews/raspberry-pi-4-b,6193.html (Accessed: 01/03/2020)

Sneddon, J (2020) *'Must Read: The Raspberry Pi 4 is Here, And it's a BEAST!'.* Available at: https://www.omgubuntu.co.uk/2019/06/raspberry-pi-4-specs-price-buy (Accessed: 01/03/2020)

The PiHut, (2020) *'Raspberry Pi Camera Module V2'.* Available at: https://thepihut.com/products/raspberry-pi-camera-module?src=raspberrypi (Accessed: 01/03/2020)

Keller Technology (2018) '7 Types of Sensors for Object Detection' Available at: https://www.kellertechnology.com/blog/7-types-of-sensors-for-object-detection/ (Accessed: 01/03/2020)

Miller, L (2019) *'IR Sensor vs. Ultrasonic Sensor: What is the difference?'.* Available at: https://www.learnrobotics.org/blog/ir-sensor-vs-ultrasonic-sensor/ (Accessed: 01/03/2020)

Telemecanique Sensors (N/A) *'Understanding the differences between Ultrasonic and Photoelectric technologies'* Available at: https://tesensors.com/sites/default/files/news/downloads/Ultrasonic%20vs.%20Photoelectric_the%20difference.pdf (Accessed: 01/03/2020)