IDP Report

Team: Wall-IDP (M110)
Members:
- Nivi Chandra Bose - Lab Group 59 - Girton College
- Vlad Filip - Lab Group 59 - Girton College
- Tungsten Tang - Lab Group 60 - Girton College
- Michaela Karassellos - Lab Group 61 - Girton College
- Lorcan Nicholls - Lab Group 61 - Girton College
- Phil Jiang - Lab Group 63 - Girton College
- Joseph Slimmon - Lab Group 63 - Girton College

Approach To The Task

Our approach to the task was to consider the major tasks or decisions and build up a concept around that. These decisions were:

→ Metal detection methods
→ How to move the blocks
→ Which path to follow around the table

We also wanted to focus on keeping the design simple. Part of this means having as few moving parts as possible. This makes it easier to program and gives less opportunity for errors or mistakes. A simple design with quick construction methods means new parts can be added on easily and we can ensure production is managed within the short time frame.

Following the initial presentation, where it was noted that we have been attempting too much and not narrowing down our focus, we have adjusted our approach to ensure that we make decisions on what exactly we want to use and achieve.

Robot Concept And Design

Concept Evaluation

Instead of designing whole robots and evaluating those robots, we focused on certain elements to design the best versions of those parts and combine them into one robot.

The first thing considered was our metal detection method. Our ideas included:

→ Magnet and Hall effect sensor
→ Shake it and use noise detection
→ A coil and emf changes
→ Weight

Shaking it was quickly rejected as in the competition there is likely to be a lot of noise around the robot that we cannot control and this would affect its ability to sort between the blocks. Weight was also rejected as the weight differences are very small. This left us to test the Hall effect sensor and the metal coil and based on the results we decided to use the metal coil.

The other major design decision was how we would move the blocks. The potential designs were:

→ Forklift style platform to lift and carry the blocks
→ A claw
→ A gate to drop around the block and push it towards the squares

While a forklift mechanism would be relatively simple, we decided against it, as getting under the blocks would be quite difficult. We considered the gate for a while but later realised a claw would better allow us to place the blocks in each square, with less risk of pushing other blocks out.

Following these decisions we put together the following initial design that we could prototype and adjust to ensure good integration between the different parts.

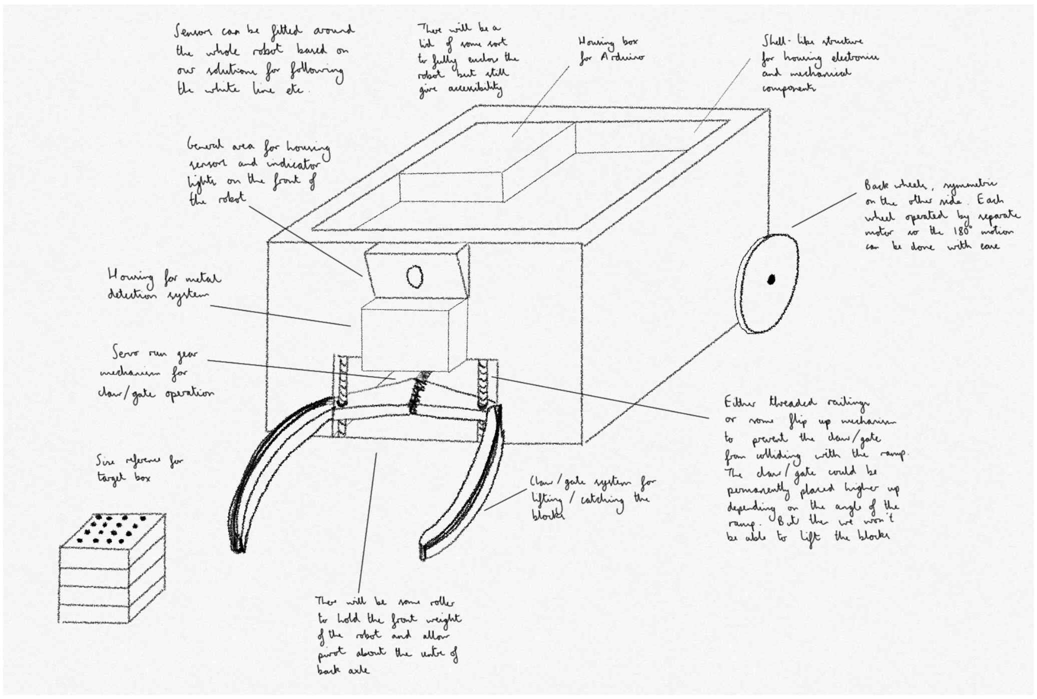

Sensors can be fitted around the whole robot based on our solutions for following the white line etc.
There will be a lid of some sort to fully enclose the robot but still give accessibility
Housing box for Arduino
Shell-like structure for housing electronics and mechanical components
General area for housing sensors and indicator lights on the front of the robot
Back wheels, symmetric on the other side. Each wheel operated by separate motor so the 180° motion can be done with ease
Housing for metal detection system
Servo run gear mechanism for claw/gate operation
Either threaded railings or some flip up mechanism to prevent the claw/gate from colliding with the ramp. The claw/gate could be permanently placed higher up depending on the angle of the ramp. But then we won't be able to lift the blocks
Size reference for target box
Claw/gate system for lifting/catching the blocks
There will be some roller to hold the front weight of the robot and allow pivot about the centre of back axle

Figure 1: Hand drawing of initial robot design

Mechanical

Our chassis design is intended to be modular. If more space is needed for sensors or electronic components, layers can be added. Each layer has a central hole to allow cables and wires to pass through to other layers.

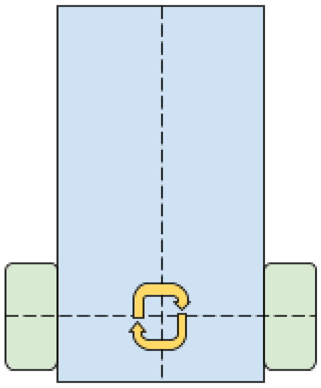


Figure 2: Diagram illustrating wheel placement and rotation expectations

Originally we were planning to place two wheels with motors at the back and a ball caster at the front. Each wheel will have its own motor. This would allow the robot to rotate about the mid-point between these two back wheels. However, upon studying the caster ball, we have determined that it is likely to get caught on the tape or the base of the ramp. This means that we have adjusted the design so that the caster ball is at the back and the wheels connected to motors will be at the front.

We have tested our robot concept with the motors and wheels. From this we have learnt that we need to stabilise the main body as the wheels did not stay aligned well, causing the robot's path to curve. Also the motors were not capable of getting the robot up the ramp. To solve this, we will attempt to use the other set of motors but if that does not provide enough power, we shall have to adjust the path to follow the walls instead.

We intend to have our robot pivot on a central point and reach the sorting squares without moving forwards or backwards, meaning our robot must be long enough to achieve this. Since the change of the wheel location this length is going to have to come from the mechanism that carries the blocks.

For this reason, we plan on using a claw mechanism to pick up the blocks and carry them to the sorting areas. Despite the complexity of the design, it should make it easier to get the blocks over the ramp and allow us to ensure that the blocks are placed within the relevant square. To prevent the claw colliding with the ramp and to lift the blocks up, we are going to use a simple raising platform to house the claw mechanism. It will consist of an MDF platform with 2 nuts embedded on either side. The platform will raise and lower through the twisting of 2 long threaded shafts via the action of a motor. The motor will be time controlled so the platform can be raised and lowered to specific heights. The preliminary design for the claw uses a servo to actuate the back gear to turn anticlockwise and close the claw. This design does take up a lot of space so a potential alternate design could be connecting the actuating gear directly to one of the claw arms and using a further gear to move the other arm. This would take up less space but be susceptible to a lot more sources of friction and have complexity in securing the individual gears. Further analysis after the first claw has been built will be done to decide whether an alternative design is required.

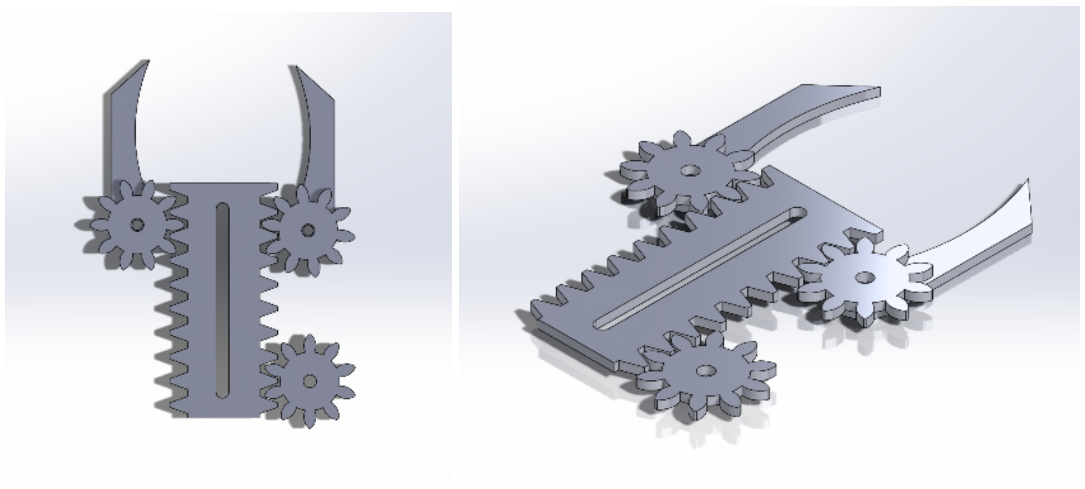


Figure 3: CAD models of claw mechanism

Construction and Materials

Most of our robot’s main body is going to be constructed from metal, specifically aluminium. This is due to the fact that metal is easy to fold into shape so we can quickly and easily add layers to our robot if and when we need more space. The aluminium is also lightweight and strong. The folded aluminium also gives a level of flexibility that means the shape can be adjusted easily if there is, for example, an alignment problem with the wheels.

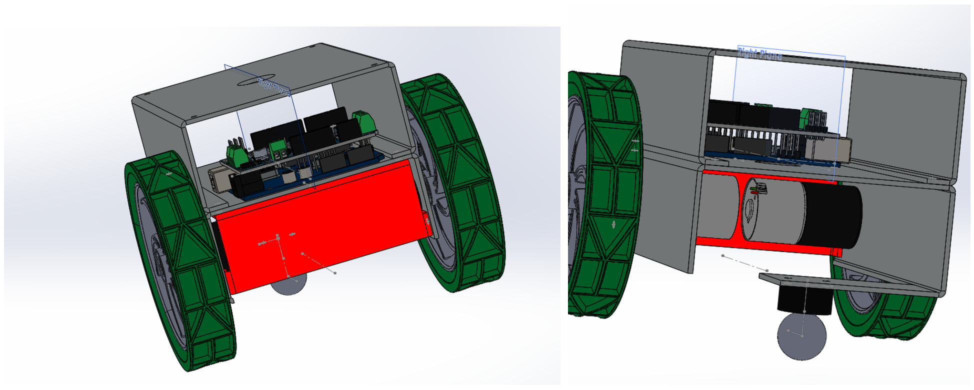


Figure 4: CAD model (SolidWorks) for 'cart' structure (chassis with attachments)

Our sensors are going to be placed at various points in and around the robot depending on where they work best. Our layered design allows us to easily add more space for sensors in our design and ensures that, while all the electric components are covered, there is still easy access for repair or improvement.

<u>Electronics/Sensing</u>

List of sensors and circuits

Following feedback from the initial presentation, we decided to test and use a smaller number of sensors and rely on existing data sheets for information in order to make our building time more efficient. This also links to the risk of having too many sensors which will make the robot too complicated with signals and inputs overlapping, causing confusion. Therefore, we have tried to consider these factors and have come up with the following required sensors and circuits:

→ Metal detector

The metal detector consists of an inductive coil that the material will pass though, an LED that will light up if metal is detected, a capacitor and resistor. Tests were done on the initial metal detector built with a 25 SWG copper coil and we concluded that it was not powerful enough to detect the block containing metal. So, we used a thicker insulated coil and increased the turns to 32, which resulted in a more sensitive metal detector that could distinguish between the block with metal and the block without metal, with some error. Therefore, we are planning to build an amplifier to amplify the output of the coil so that it is more easily detected by the circuit, causing the LED to light up. The circuit is also really sensitive to motion and other metal objects nearby, so the next steps will have to be to consider how to solve this problem with the Mechanics team.

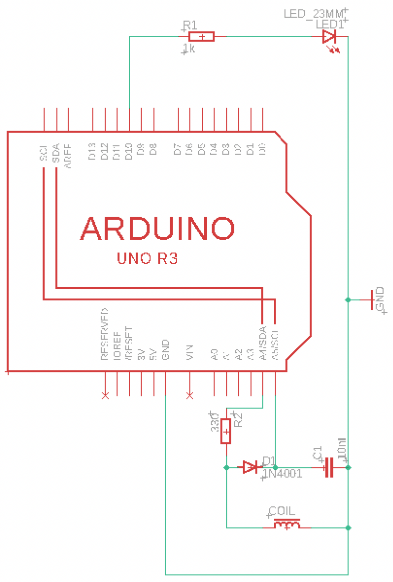


Figure 5: Diagram of Arduino

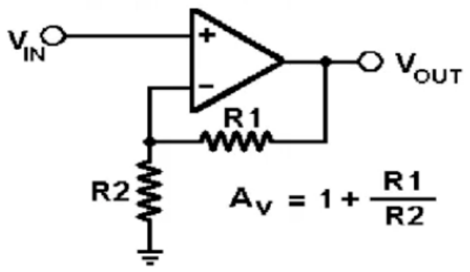


Figure 6: Non-inverting amplifier circuit using op-amp

→ Colour sensor

This will be used to determine whether the robot has reached the correct delivery area.The main approach to creating the colour sensor is to use one white LED to flash light onto the environment which will then be reflected onto a LDR connected in parallel. This will detect the presence of different colours and output a voltage to the Arduino which can be calibrated according to the reflectance to differentiate between colours based on their output ranges. The colour sensor will be used to distinguish between the red and blue squares in the delivery area. We will also use an op-amp to make a comparator circuit to convert this analogue input to a digital input. Whilst testing, we found that the colour sensor worked well for the red colour detection but was less accurate to detect blue. So, we are going to look into adjusting the resistor in the potential divider to improve sensitivity.

Testing the circuit showed a range of 1.2-1.4 for red colours and 1.6-2 for blue. These were calibrated during testing and integrated into the arduino code.

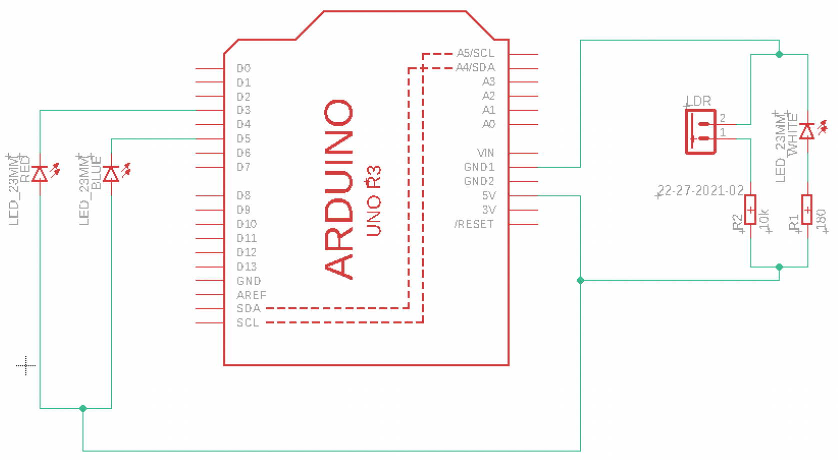


Figure 7: Colour sensor circuit and connections to the arduino

→ IR sensor

- We tested the IR sensors (connected to the arduino) and decided to use 2 sensors at the bottom of the chassis so that they are on either side of the white path line. The sensor will detect the colours of the track (black or white) and constantly feedback to the system. It should always detect black (no obstacles), but when it does detect white (obstacle), it will feed this back to the system and it will be known that it has started to go off track. This constant feedback of information can be used by the software to make decisions. There were slight delays at first with detections of the black and white colours but we found an optimum position for it to work efficiently and will have to work with the Mechanics team to make it feasible.

→ LEDs for indication of motion, actions and communication
  - We need to make sure to search up the specifications for the diodes and LEDs to make sure we connect resistors with appropriate values.

→ Decoupling capacitors to remove AC bias from the power supply and to reduce the likelihood of inaccurate sensor readings, e.g. for the metal detector.

→ An encoder can be used to measure the distance traveled by the robot and keep track of where it is along the journey to make sure it does not stray away from the path.

→ Switch debouncing (using a low pass filter) is required as the Arduino runs at a very high clock rate and so can pick up switch bouncing quite easily.

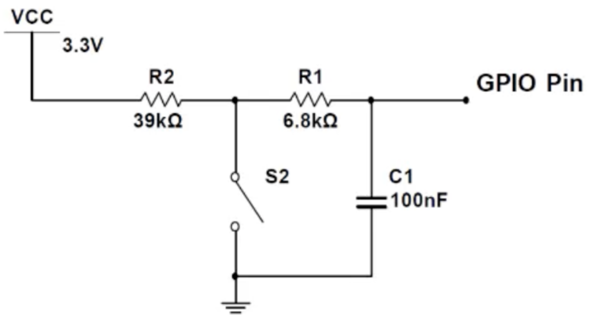


Figure 8: Switch debouncing circuit

Data processing by electronics

In order to simplify the interface between the hardware electronics and software, we decided to do some processing in the electronics as opposed to the software. For the sensors, comparator circuits with op-amps such as the Schmitt trigger will be used to convert analog input signals to digital output signals to be readily and easily used by the software.

Interfaces between different parts of the system

The arduino will act as the intermediary between the remote computer and the robot-driving motors. It is responsible for relaying the electrical signals from the sensors (i.e. cameras, magnetic detectors and such) to the Motors. It also communicates with our remote computers through a bluetooth connection. The aim is to be able to send commands to the Arduino to indicate our desired trajectory of motion. The Arduino will then translate the information into commands for the motors, as well.

The Arduino will also send sensor data to the remote computer, coupled with the live stream data from the Overhead Cameras (a connection facilitated by the use of WiFi via PuTTy) will be able to navigate the table and ramps through the use of computer vision and Python Functions.

Our vision is to have the computer drive the Arduino. The computer (using Python) will perform the more complex procedures, namely operating computer vision packages like OpenCV to accurately distinguish between the white line and black table, and the Arduino will perform the simpler tasks locally to run the motors accordingly.

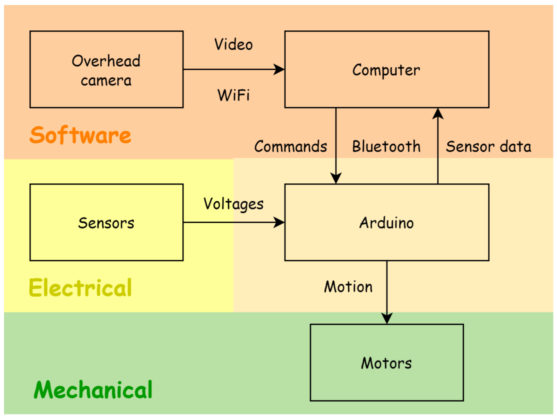


Figure 9: Diagram showing integration needed between different elements

<u>Software</u>

Exploration and Navigation Algorithms

Given the aforementioned issues we ran into, the specifics of the navigation algorithm has not yet been decided. However the initial plan was to have the robot follow the white lines and when reaching the vicinity of the blocks, it would pick one (pre-decided) and test. Once tested, it will then make the decision to pick that box up or the other, then return to the end location again via the white line.

Risks And Challenges

When deciding whether to utilise bluetooth technology as opposed to WiFi for communication between the computer and the Arduino, we identified and compared what we considered to be the most important factors; latency, simplicity to implement and limitations with regards to space/distance.

However we discovered almost immediately that there was an inability for the arduino to communicate through the ESP32 package correctly. We found that the updated ESP board manager (v2.6.1) was too new for the arduino which required ESP board manager v1.0.6), requiring us to downgrade our board manager to match the Arduino, which was a very time consuming process.

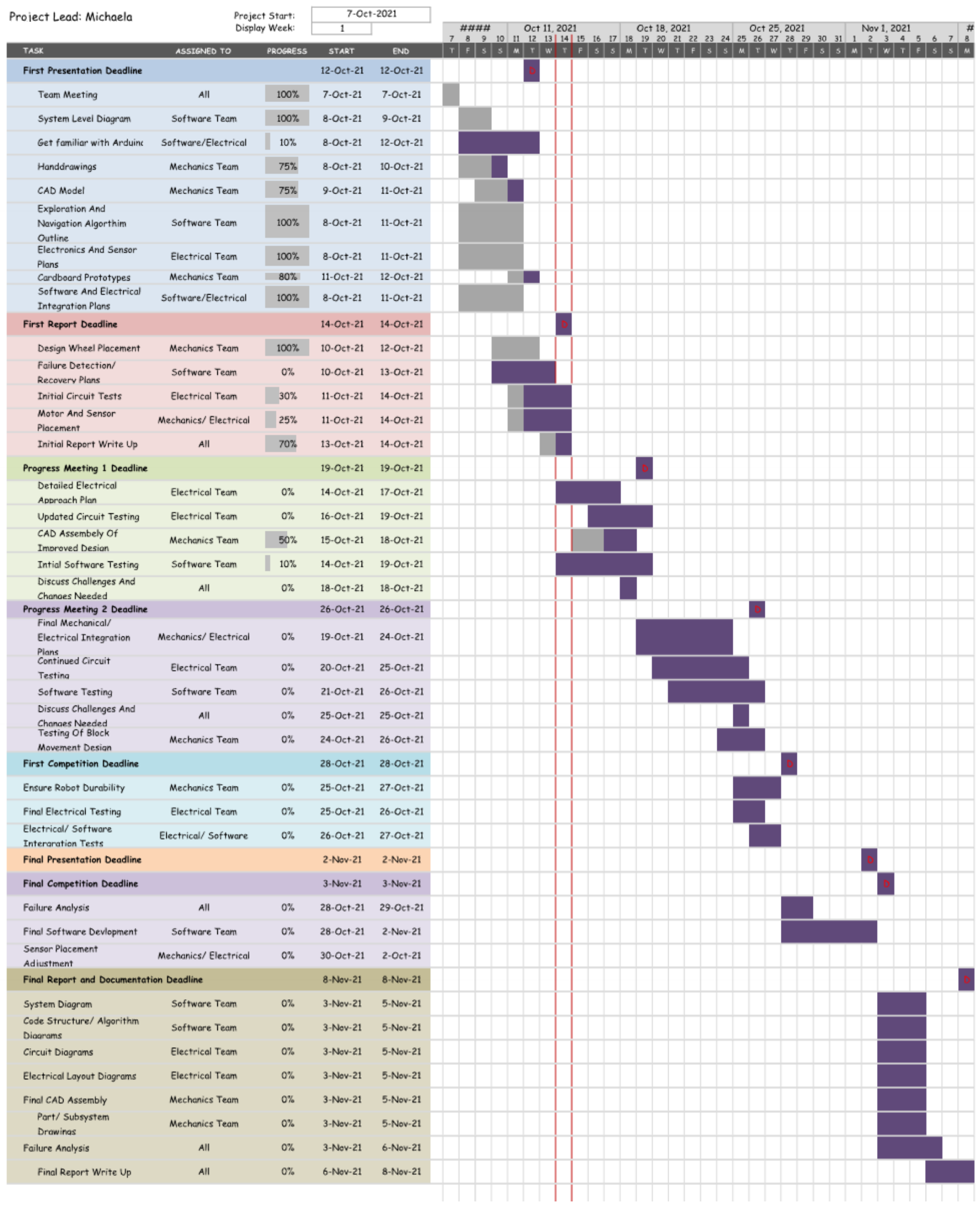
The biggest challenge that we have encountered is setting up the communications between the different systems of our robot. Given the lack of information for Bluetooth and WiFi set up, or the incompatibility of the software and hardware and our first Arduino board had issues that required us to get a replacement all of these obstacles posed challenges to our aim of staying on schedule. We anticipate future technical issues when using Bluetooth and WiFi and will plan to budget more time and resources to allow for considerations in the extra effort that will be potentially required to have these connections up and running.

Some more specific challenges we were anticipating was how to set the correct threshold voltage to be able to filter out digital or magnetic noise to accurately identify whether the robot has located not only the correct block, but whether the signal its reading is actually a block.

Another challenge is to utilise cameras to identify the white lines from the black table, and to follow it. By passing through camera data through the WiFi connection, the camera will send information to the arduino (driven by Python) and the arduino will make adjustments to its trajectory (via the navigation algorithms that we will implement; i.e. adjustments to its angle and yaw, velocity of wheels etc). One of the biggest challenges that we identified is to learn and become proficient enough in OpenCV (a computer vision package that is offered through Python) to code a program robust enough to reliably detect the white lines.

Resource And Time Allocation

We are using a Gantt chart to ensure good time management. Each deadline and progress meeting is classed as a phase on it, allowing us to see if we are going to keep up with each deadline.



Project Lead: Michaela

Project Start: 7-Oct-2021
Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END
First Presentation Deadline			12-Oct-21	12-Oct-21
Team Meeting	All	100%	7-Oct-21	7-Oct-21
System Level Diagram	Software Team	100%	8-Oct-21	9-Oct-21
Get familiar with Arduinc	Software/Electrical	10%	8-Oct-21	12-Oct-21
Handdrawings	Mechanics Team	75%	8-Oct-21	10-Oct-21
CAD Model	Mechanics Team	75%	9-Oct-21	11-Oct-21
Exploration And Navigation Algorthim Outline	Software Team	100%	8-Oct-21	11-Oct-21
Electronics And Sensor Plans	Electrical Team	100%	8-Oct-21	11-Oct-21
Cardboard Prototypes	Mechanics Team	80%	11-Oct-21	12-Oct-21
Software And Electrical Integration Plans	Software/Electrical	100%	8-Oct-21	11-Oct-21
First Report Deadline			14-Oct-21	14-Oct-21
Design Wheel Placement	Mechanics Team	100%	10-Oct-21	12-Oct-21
Failure Detection/ Recovery Plans	Software Team	0%	10-Oct-21	13-Oct-21
Initial Circuit Tests	Electrical Team	30%	11-Oct-21	14-Oct-21
Motor And Sensor Placement	Mechanics/ Electrical	25%	11-Oct-21	14-Oct-21
Initial Report Write Up	All	70%	13-Oct-21	14-Oct-21
Progress Meeting 1 Deadline			19-Oct-21	19-Oct-21
Detailed Electrical Approach Plan	Electrical Team	0%	14-Oct-21	17-Oct-21
Updated Circuit Testing	Electrical Team	0%	16-Oct-21	19-Oct-21
CAD Assembely Of Improved Design	Mechanics Team	50%	15-Oct-21	18-Oct-21
Intial Software Testing	Software Team	10%	14-Oct-21	19-Oct-21
Discuss Challenges And Changes Needed	All	0%	18-Oct-21	18-Oct-21
Progress Meeting 2 Deadline			26-Oct-21	26-Oct-21
Final Mechanical/ Electrical Integration Plans	Mechanics/ Electrical	0%	19-Oct-21	24-Oct-21
Continued Circuit Testing	Electrical Team	0%	20-Oct-21	25-Oct-21
Software Testing	Software Team	0%	21-Oct-21	26-Oct-21
Discuss Challenges And Changes Needed	All	0%	25-Oct-21	25-Oct-21
Testing Of Block Movement Design	Mechanics Team	0%	24-Oct-21	26-Oct-21
First Competition Deadline			28-Oct-21	28-Oct-21
Ensure Robot Durability	Mechanics Team	0%	25-Oct-21	27-Oct-21
Final Electrical Testing	Electrical Team	0%	25-Oct-21	26-Oct-21
Electrical/ Software Interaration Tests	Electrical/ Software	0%	26-Oct-21	27-Oct-21
Final Presentation Deadline			2-Nov-21	2-Nov-21
Final Competition Deadline			3-Nov-21	3-Nov-21
Failure Analysis	All	0%	28-Oct-21	29-Oct-21
Final Software Devlopment	Software Team	0%	28-Oct-21	2-Nov-21
Sensor Placement Adiustment	Mechanics/ Electrical	0%	30-Oct-21	2-Oct-21
Final Report and Documentation Deadline			8-Nov-21	8-Nov-21
System Diagram	Software Team	0%	3-Nov-21	5-Nov-21
Code Structure/ Algorithm Diagrams	Software Team	0%	3-Nov-21	5-Nov-21
Circuit Diagrams	Electrical Team	0%	3-Nov-21	5-Nov-21
Electrical Layout Diagrams	Electrical Team	0%	3-Nov-21	5-Nov-21
Final CAD Assembly	Mechanics Team	0%	3-Nov-21	5-Nov-21
Part/ Subsystem Drawings	Mechanics Team	0%	3-Nov-21	5-Nov-21
Failure Analysis	All	0%	3-Nov-21	6-Nov-21
Final Report Write Up	All	0%	6-Nov-21	8-Nov-21

Figure 10: Gantt Chart (https://universityofcambridgecloud-my.sharepoint.com/:x:/r/personal/mak93_cam_ac_uk/_layouts/15/guestaccess.aspx?e=X5Pyyp&share=EWA_DqTTfzhPrpPOTkExLBkB4-Bu0JEMp9lhDBHLFXZZYw)