Imperial College London

EEEBalanceBug Report

ELEC50008 - Electronics Design Project 2

20 June 2023

Mr Hakan Merdan
Dr Esther Perea

Group 13
Oliver Cosgrove - CID 02043827
(MEng Electronic and Information Engineering - Year 2)
Patrick Beart - CID 02061413
(MEng Electronic and Information Engineering - Year 2)
Corey O'Malley - CID 02050283
(MEng Electronic and Information Engineering - Year 2)
Jacob Alexandrou - CID 02037266
(MEng Electronic and Information Engineering - Year 2)
Tina Dionysiou - CID 02094746
(MEng Electrical and Electronic Engineering - Year 2)
Marta Lopez Gallo - CID 02088083
(MEng Electrical and Electronic Engineering - Year 2)

Word count: 10,631

Abstract

This report addresses the development of an automated self-balancing charting rover, which employs computer vision to map a maze by locating its relative position through resection, making use of three coloured beacons and ultimately, finding the shortest path through the maze. To tackle the goal there were four main components: ensuring the DC grid kept the beacons constantly powered up to allow the rover to accurately locate itself at any instant. Secondly, ensuring the rover could freely navigate the maze while mitigating the risk of falling over. Also, ensuring computer vision correctly calculates position in the maze, and then the server correctly translates this information to draw the map of the maze, and guide the rover to its next location. A robust final design of the rover's structure was produced using CAD, 3D printing and laser cutting, which successfully supported all the electronics needed to obtain the information from the maze and communicate it to the server. After thorough testing and upon integration of all sub-modules, a successful final model was developed, which fulfilled all design specifications whilst remaining under budget.

Contents

1 Introduction

1.1 Description

The aim of this project is to design and build an autonomous balancing rover which is able to navigate a maze whilst mapping it.

The design of the rover had to adhere to the design specifications given by the customer, which can be summarised as the rover being able to complete three main tasks. The first task is to be able to move autonomously through a maze without crossing its boundaries. The second task entails the rover being able to autonomously analyse the layout of the maze and calculated the shortest path from the start to the finish of the maze. Finally, the third main task is for the rover to move autonomously on two wheels, for which a controller system was developed to balance the rover and allow it to move in response to position and heading commands.

In order to be able to complete these tasks the team had draw from theoretical and practical knowledge gained from the academic year, as well as external sources to apply them to a project.

The design of the rover was subject to constraints, both functional and non-functional, which are described in section 1.2 (Project Design Specification). The actual design process is described in part 3 (Design Process).

1.2 Project Design Specification

1.2.1 Requirements

The requirements that the rover must full-fill can be split into two sections: functional and non-functional requirements.

The functional requirements are: the rover must be able to move through the maze without crossing the illuminated boundaries as well as survey the layout of the maze, producing a map of it overlaid with the position of the robot and calculate the shortest path through the maze. All of this must be done autonomously. In addition, the rover must be able to balance on two wheels and complete the rest of the tasks without falling over.

The non-functional requirements include the rover being able to complete the task with no intervention reliably. Its construction must be robust and efficient. Furthermore, it must be coded for usability, testability, maintainability and scalability.

1.2.2 Environment

The environment in which the rover will be tested consists of an artificial maze made out of light strips on a black arena surrounded by black curtains. Figure 1 shows what the maze is expected to look like. The maze will be placed inside the EEE Department level 1 lab, making the expected temperature on the day of the demo to be around 25^{o}C. This means that there should not be any interference on the rover from temperature, as all components are expected to perform well within these ranges.

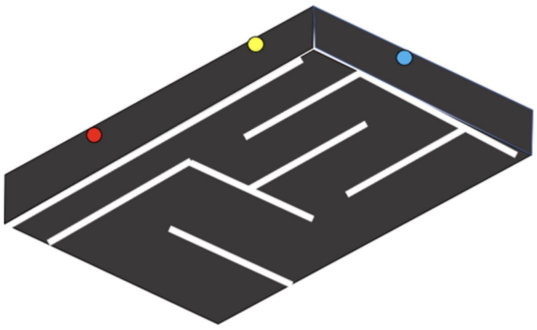

Figure 1: Maze Diagram

1.2.3 Target Cost

The budget allocated for the rover is limited to £60. It is expected that the whole budget will not be used.

1.2.4 Customer

This product's customers are Mr Hakan Merdan and Dr Esther Perea Borobio, although it is coded for scalability.

1.3 High Level Design

Figure 2 shows a high level view of how the design is split. The energy system, which powers the beacons, is independent from the rover and therefore is not integrated with the rest of the rover.

The rover system consists of three main subsystems: the first is the Command and Control (C2) system, which consists of a C++ program running on an Espressif ESP32 microcontroller. This subsystem is responsible for carrying out the process of navigating the maze: it instructs the drive subsystem which angle and displacement to target, takes in information from the Light Dependent Resistors (LDRs) and detects turning points, where it carries out the turning procedure. Subsequently it configures and receives data from the camera subsystem to determine its position, and then sends and receives information from the webserver on where to go next.

The drive subsystem is responsible for running the control system to balance the rover, taking in commands over UART to control the desired position and heading and counting and sending back commanded motor steps to determine the rover's position in between beacon location procedures. It communicates with an MPU6050 over I^2C to determine its orientation and acceleration, then feeds this into a cascade control structure to control the step signals sent to the motor control board.

The vision subsystem consists of a NIOS II soft-CPU and an RTL design both running on an Intel MAX10 FPGA: the RTL design reads in pixels from the D8M-GPIO camera module and, based on the configured target colour and other parameters, delivers the beacon position and confidence, which the soft-CPU reads and forwards to the C2 ESP32. The soft-CPU also configures camera parameters such as gain, exposure, focus and zoom over a MIPI interface.

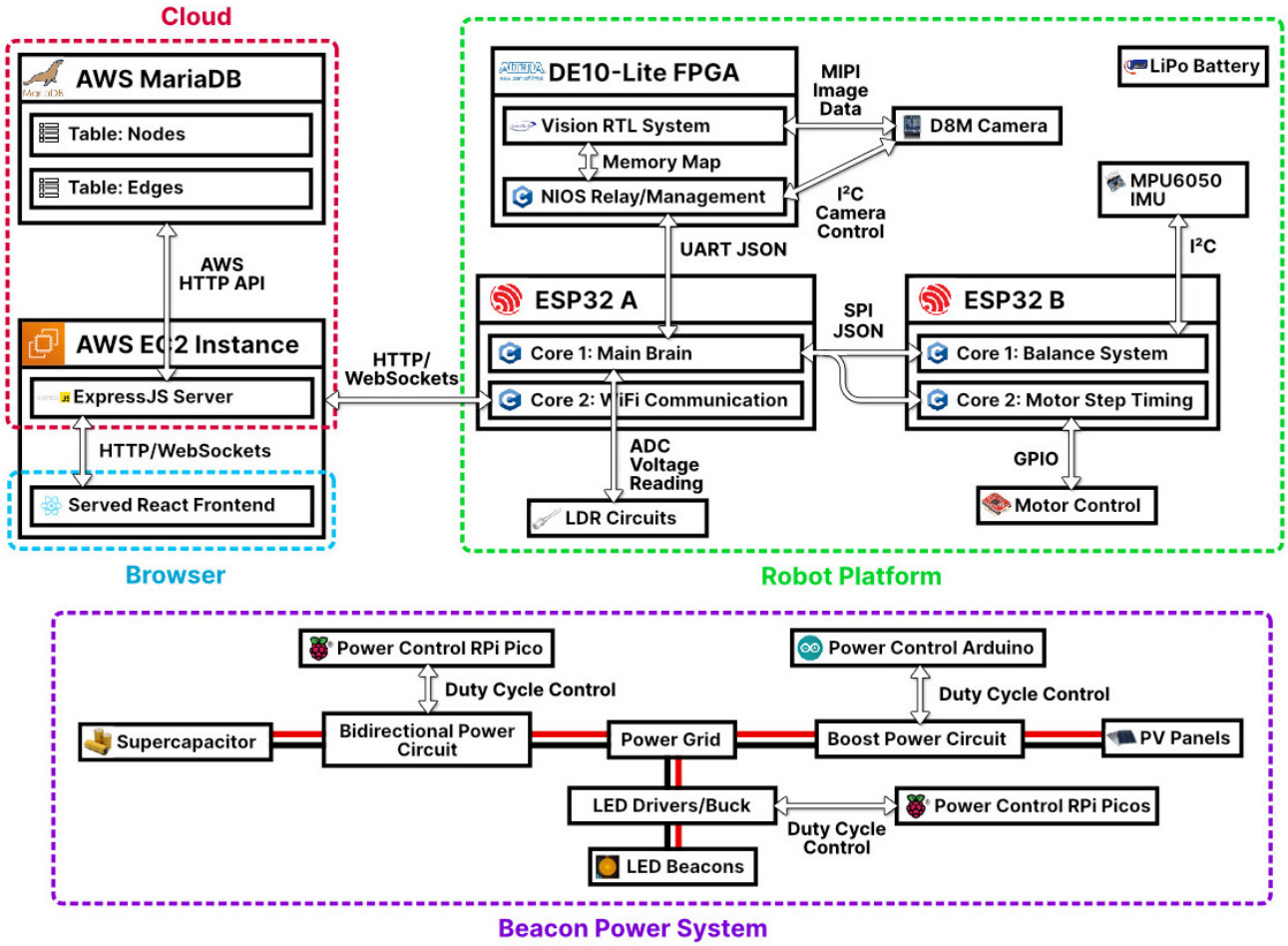

Figure 2: High Level Description of Rover System

2 Project Management

2.1 Work Breakdown Structure

The technical workload was divided into six sub-modules: control, energy, vision, networking, design and integration. Figure 3 shows how each of these sub modules was broken down into the unique components to be able to create the respective sub module. This was done at the beginning of the project by the whole team as a way to break down the project into smaller, more manageable elements.

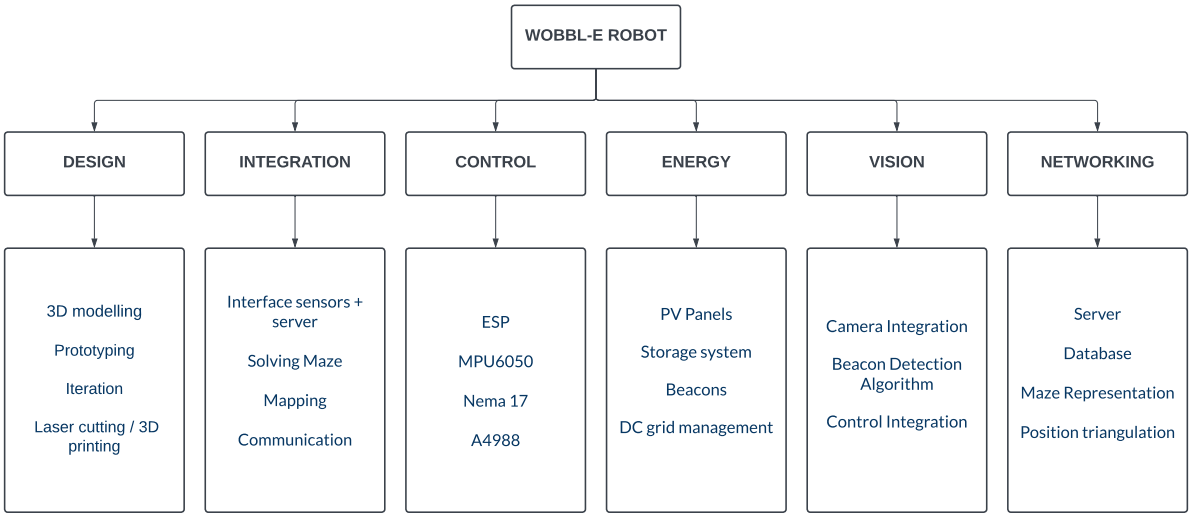

Figure 3: Work Breakdown Structure Diagram

2.2 Task Allocation

To allocate the tasks fairly, the workload was divided both in terms of amount and knowledge requirements. Firstly, the team decided that to approach the project efficiently it was better to tackle the submodules shown in the Work Breakdown Structure separately and later integrate them. Each submodule was researched thoroughly be each person/people allocated to it, using knowledge obtained during the year and external resources. This was done to obtain a holistic view of each submodule, as well as to further break them down into less complex tasks.

The energy system and circuits needed for the vision system were tackled by both Tina and Marta, being EEE students. Since the rest of the subsystems required more programming knowledge, they were assigned to EIE team members. On the hardware aspects of the rover the control system for the balancing of the rover was taken on by Oliver and the vision system was designed and implemented by Patrick. Regarding the software aspects of the project, the communication and networking systems were tackled by Jacob and Corey respectively. The design of the rover was discussed as a team throughout and modelled by Jacob and Oliver.

When these sections were completed, the whole team came together to assemble and test the rover. Jacob was in charge of the actual design of the rover, as he had previous experience with the tools necessary. Firstly, Jacob, Corey, Patrick and Oliver took part in integrating all the components of the rover. As mentioned previously, the energy system is separate from the rover, so it could be tested first as a separate entity.

Finally, all the different components were brought together and tested as a whole.

In terms of the non-engineering tasks, Marta was in charge of writing and editing the report, with Tina as a second editor.

2.3 Team Meetings

The team met almost daily in the lab during week days and set up longer weekly meetings to discuss the progress that had been done in the previous week as well as the next steps to be completed. This allowed all team members to better grasp the progress of the project as a whole as well as how everything was interconnected. These meeting were also a chance to address any issues that someone needed help dealing with. The team also set up a group chat to facilitate communications when they were working separately.

2.4 Project Timeline

The completion of the project had to be achieved in five weeks. In order for the whole team to be aware of what was happening and to prepare for any unprecedented circumstances, a sensible timeline for each component of the project was essential. An initial draft of this was developed using Notion, which was constantly updated by the team members to adjust to how the project was progressing.

Figure 4 shows a Gantt chart with the final timeline of the project, after the final updates before the delivery of this report. The high level tasks to be completed are found on the left of the figure and the duration of each of them is shown by the blue boxes.

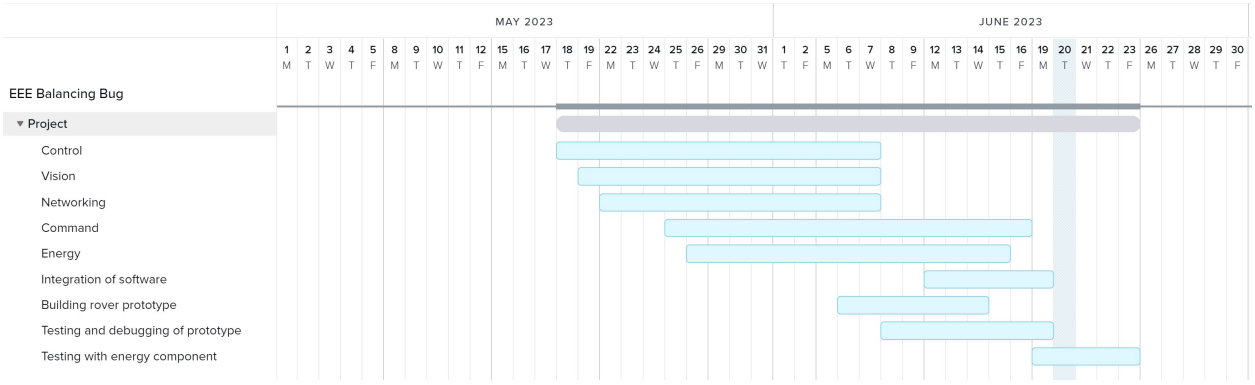

Figure 4: Gantt Chart

3 Design Process

3.1 Control

To begin with, a thorough analysis of the rover's physics was conducted. This involved sketching the rover and examining its system dynamics. By modeling the physics of the system, it was determined that controlling the angular acceleration of the wheels would enable the manipulation of all relevant variables governing the rover's movement [1].

Building upon this understanding, the control structure for the rover was devised. Drawing inspiration from the drone lab, a similar cascade control structure was implemented. As the rover had not yet been received, a MATLAB model was created in Simulink. The cascade control structure, shown in figure 7, consisted of multiple loops, with the output being the angular acceleration of the wheels. The inner loop utilized a PID controller to regulate the angle rate, followed by a proportional controller to govern the pitch. This is shown in figure 5. The subsequent loop controlled the velocity of the rover using a PID controller, with a proportional term applied to the position. This is shown in figure 6.

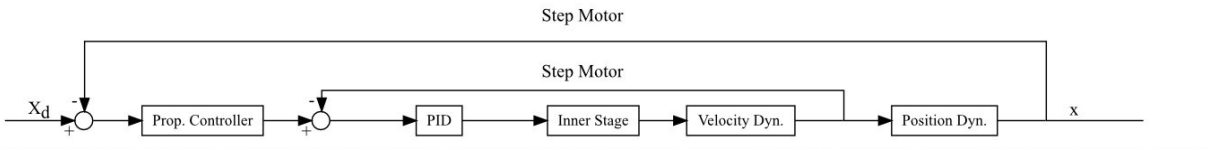

Figure 5: Pitch Controller

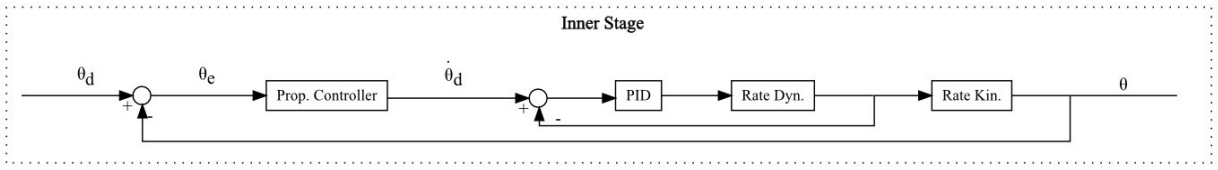

Figure 6: Position Controller

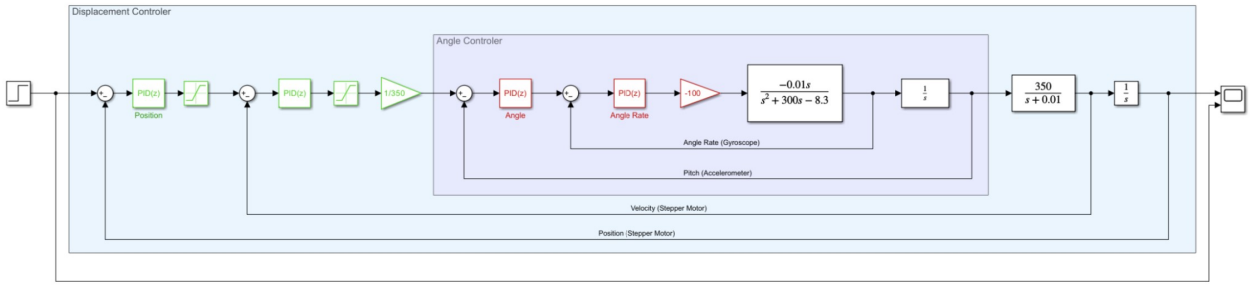

Figure 7: Cascade Control Structure Simulink Model

To effectively control the motors, two distinct tasks were identified. Task 1 involved fetching data from the sensor and running the controller, while Task 2 focused on utilizing the obtained angular acceleration to

step the motors accordingly. To ensure optimal performance, it was crucial to address the time constraints imposed by sensor data retrieval. As this process required approximately 2.5ms and considering the 16x microstepping, the motors' maximum speed was deemed too slow, necessitating the execution of the tasks on separate cores of the ESP.

In order to filter the sensor data, a Kalman filter [2] was implemented. For full calculations see appendix A. The Kalman filter design involved utilizing the Kalman equations and incorporating inputs from the gyro (angle rate) and the accelerometer (pitch) obtained from the MPU6050 sensor. The filter's outputs were pitch and pitch rate bias, which further aided in accurately controlling the rover.

For the sake of flexibility and ease of modification, a PID class was created. This allowed for the straightforward adjustment of PID values and facilitated the transition to the next state of the discrete controller. Initially, a continuous-time PID controller was designed, which was later transformed into discrete-time using the backward Euler transformation. For full calculations see appendix B. This approach allowed the sample time to remain variable, providing the flexibility to adjust the controller's operating frequency.

Following the implementation of the control system, extensive tuning of the PID values for the angle rate and pitch controllers was carried out. This tuning process resulted in successful balancing of the rover, enhancing its stability and control.

Subsequently, attention was shifted towards tuning the velocity and pitch controllers, incorporating saturation limits to prevent the rover from tipping over. By fine-tuning the PID values and imposing restrictions on the desired angle and velocity, the control system effectively managed the rover's movement.

To enable turning maneuvers, a differential stepping strategy was implemented. This involved staggering the motor steps and controlling the angular acceleration difference between the wheels using a simple angle and angle rate cascaded control structure. Further refinement was achieved through PID tuning and the application of saturation limits to the rover's angle rate.

In summary, the iterative process of designing and implementing the control system involved analyzing the rover's physics, developing a cascade control structure, filtering sensor data using a Kalman filter, implementing a PID class for flexibility, fine-tuning PID values, and incorporating saturation limits. These steps collectively resulted in a robust control system capable of successfully balancing and maneuvering the rover.

3.2 Integration

The rover carries a central ESP32, which commands other modules and is responsible for low-level rover navigation and communication with the server. It communicates with the camera subsystem NIOS II CPU over UART, with the motor controller ESP32 over SPI and with the web server over HTTP, using a REST API as well as WebSockets. In general, the central ESP32's software is designed to be as robust as possible: it is resistant to packet corruption and loss by virtue of all communication links not relying on single instruction packets to change behaviour, but rather sending the desired subsystem state (and receiving the subsystem's state) at fixed intervals, on the order of hundreds or thousands of Hertz, so that the collection of systems never finds itself desynchronised.

It is also built with a robust system of monitoring, including warning and error states, shown in Figure 8, and the ability to reboot each subsystem individually through connections to microcontroller RESET and ENABLE pins on each subsystem: for example, if the C2 ESP32 does not receive any valid JSON communication from the camera subsystem for more than a certain amount of time, it first outputs a warning in the console and then attempts to reset the subsystem by pulling its RESET pin high for a few milliseconds and then low again, pausing system operation until this is done and the subsystem has successfully rebooted.

Figure 8: Screenshot of Command and Control ESP startup process and state debug output

The command and control ESP operates according to a simple state machine, shown in figure 9.

When it encounters a vertex (defined by a "non-corridor" reading on the LDRs, where there is either a light wall in front, or not a light wall at both sides), it performs a turning procedure to locate each beacon and determine the angle between them, then combines these angles with its knowledge of the position of each beacon within the maze to determine an estimate of its position and heading, which it combines with the inertial estimates obtained from the stepper motors to obtain a new position estimate. It then contacts the server, informing it of the vertex and the possible paths it could take (determined from LDR readings taken as it turns). The server then checks this vertex against other vertices in the database to see if it matches an already-encountered one, and instructs the rover which heading to travel along to avoid retreading its steps, and find the least explored parts of the maze.

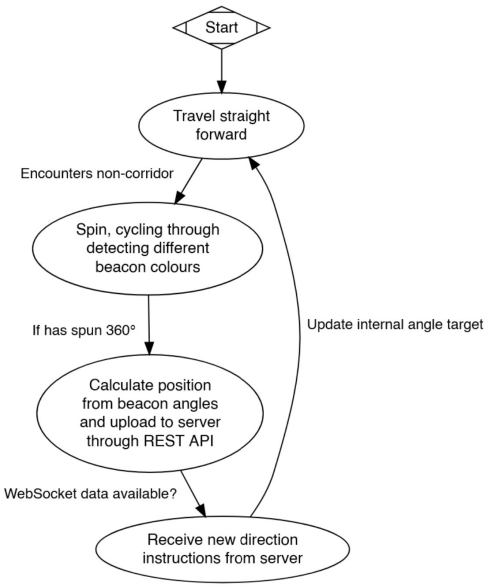

Figure 9: Command and Control overview state diagram.

The maze markings are detected using LDRs: they connect to a board which connects to the central ESP and provides a potential divider circuit for each LDR, to enable a voltage from 0-3.3V to be read to determine the brightness measured by each LDR. The circuit diagram for the LDR circuitry is shown in figure 10.

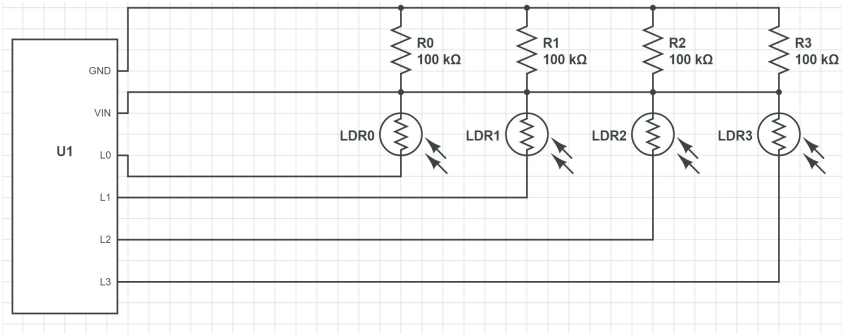

Figure 10: LDR maze detection circuit diagram

3.2.1 Communication

For communication many different factors led to the use of an array of different standards for different reasons. First HTTP was setup as it has been used for REST API control for many years and is as such easy to work with and first develop when creating the web components.

HTTP works very well for the node and path data as it is information sent infrequently and stably to the server that then handles all the database processing and position information. But this becomes worse for more frequent data requests, as such a connection orientated standard was found to be effective and decided to use web sockets for this, as the HTTP server was already created we could also “piggy back” the web socket connections onto this server allowing for ease of design.

The two web socket connections each transfer low latency information through the server as an intermediary, first the client can use manual control if needed with a web socket sending control straight to the Esp

Control, or position data can be sent to the client to be rendered efficiently without database storage.

Additionally for local connections between the two Esps the design needed to add a protocol that would quickly send data between each other with low latency and overhead as such to not interfere with either the processing of the master Esp or the Drive Esps Control System. Therefore to combat this the creation of a custom SPI library that uses the built in Esp32 Drivers [5] allows for extremely fast communication. This also included easy to use buffers for multiple transaction queuing as well as callbacks to make sure message handling is completed immediately on the masters request. The final communication standards used are shown in figure 11.

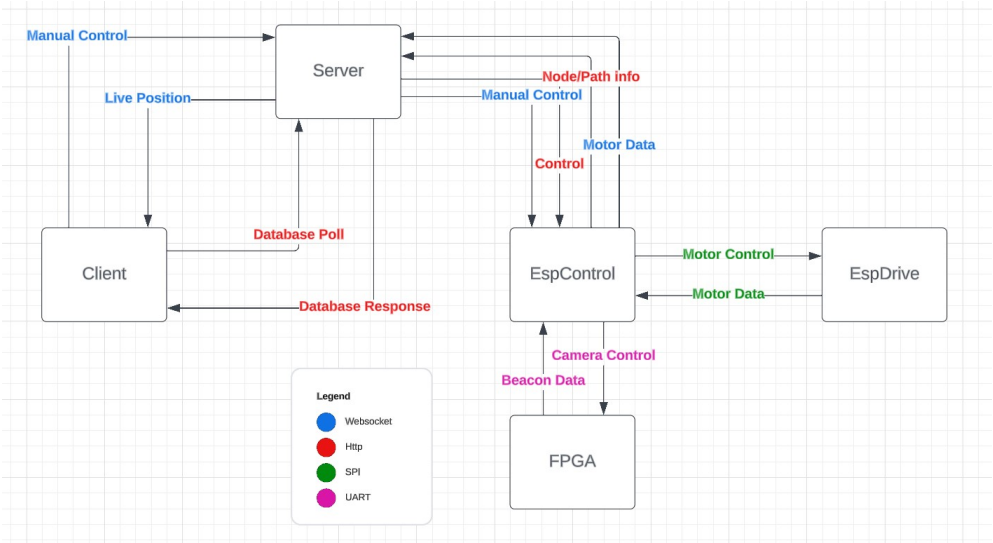

Figure 11: Communication Standards

3.3 Networking

3.3.1 Overview

The web portion of the project is important as it acts as the front facing component of the rover with how a user may view the mapping. Additionally it does a large amount of heavy lifting in terms of position calculation and algorithmic path finding. To accomplish this 3 components were developed, the server, client and the database, which work together to form the web section of the project.

3.3.2 Server

There were many choices that needed to be made for designing how each of the web components integrated with each other, as well as which libraries should be used to allow this to be done effectively. For the server, the use of ExpressJS was chosen to allow development of a HTTP server with ease, and to allow easy integration with MySQL queries, making the REST API quick to develop and prototype. Hosting was also simple as we used AWS to make this quick and easy. Additionally the use of Node means many packages are readily available to allow for further complexity such as using a WebSocket library to create two WebSocket servers attached to the HTTP server, which allowed us to develop more efficiently and in parallel as a team.

3.3.3 Database

The choice of MariaDB for the SQL server was very simple, it is quick to setup up as a local running server on the AWS instance and as such works with MySQL queries from the server without much hassle.

Laying out the database format took little time as the given data had 2 distinct sections, Node information and Path information. The Paths and Nodes were linked, with multiple paths able to connect to a single node. The layout is shown in figure 13.

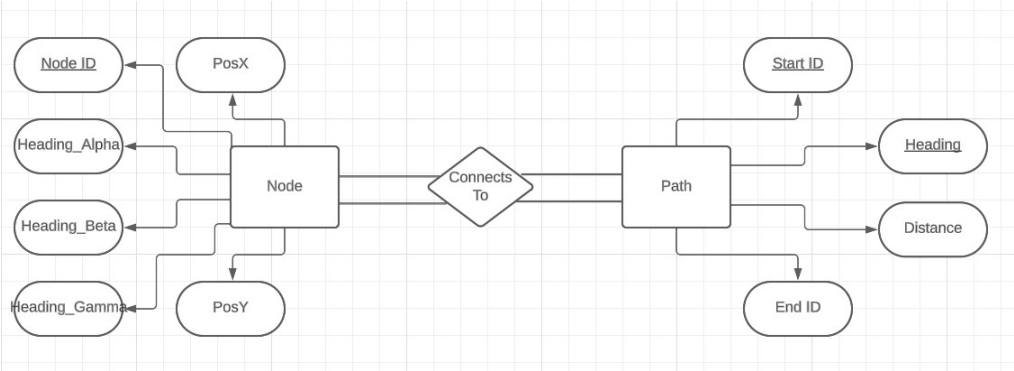

Figure 12: Database Layout

3.3.4 Client

The client renders using the React framework, the use of the virtual DOM gives much faster loading times when creating new nodes and path connections as full page reloads are not necessary. The client also had to be able to poll the database periodically to see if any changes have occurred on the server side, to do this the polling components are used. One small issue that was found was that if a path were to be updated before its connected node was, this would cause a rendering issue in which the path would not connect to the node as it had not been rendered yet, as such we call the polling of nodes at a higher rate to that of the paths to stop this from occurring.

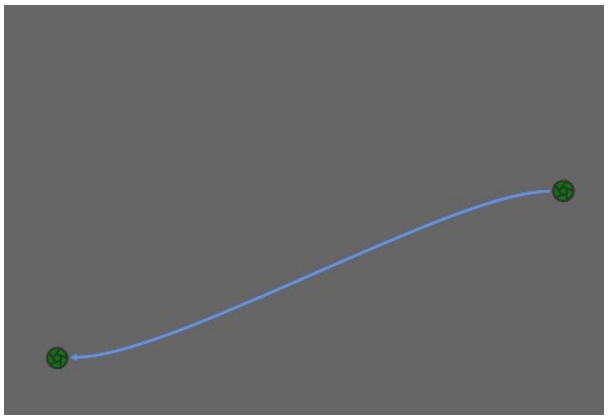

Figure 13: Maze Node Connection

3.3.5 Three Point Resection Algorithm

After analyzing many different algorithms for calculation of position using beacon angles, Tienstra's formula was a clear choice [3]. Commonly used for land surveying, it is simple to implement using basic maths to calculate position as well as only having relatively minor caveats for our purposes. It works by calculating the

difference in cotangents between the 3 known beacon positions and the angles formed between the beacons, to create scalars that, when applied to the beacon co-ordinates, can be used to find the rovers position.

For the caveats, it will always give 2 position readings for given positions that are not within the triangle formed by the three beacons, and additionally the formula will slowly lose accuracy as you move closer to the edge of the circle that the 3 points form. This is due to the exponential nature of the tangent function and the limited accuracy of the beacon readings have, as well (in extreme cases) as the accuracy of the floating point maths used to calculate these values.

First, to eliminate the 2 readings problem only values that are within the circle that the three beacons form are taken. This always resolves this issue as the two points always fall with one within the circle, and a second at the mirrored position outside of the circle.

The second issue was much harder to address as the accuracy of the beacon readings cannot be improved in this component, along with the tan accuracy. To partly resolve this, motor step counts are taken to give estimates about the position of the rover, and these integrating estimates are relied upon more when approving the edges of the maze.

Using these 3 parts we can effectively calculate our position with relatively little error or inaccuracy (see Evaluation section.)

3.3.6 Path Finding Algorithm

To find an efficient way of traversing the maze a Shortest Path Tree must be created, this allows the shortest route between two nodes to be quickly determined. There are many different options of algorithms which could have been selected to do this, including Dijkstras, Kruskals and Primms. After some discussion the choice of Dijsktras was apparent, as it is very simple to implement and easily updatable: the rover could create an SPT and update it periodically with each new added node and vertex.

To do this, the creation of a constant list of all nodes within our SPT with the shortest path to it from every other connection allows for quick path finding from the set start node to any other node with a list of all the travel weights. Furthermore we can store a full list of all the past nodes that are used to create the SPT, giving a traversable list of viable paths.

3.4 Vision

3.4.1 Prototyping

Before the camera module was set up, a Java Processing [6] application was developed to explore different ways of detecting blob-like shapes (such as the beacon lights), in an environment which permitted more rapid iteration and testing. In particular, the idea of reducing the impact of lone colour-matching pixels was investigated: the primary method considered was the application of a ‘kernel’ to the image, by which each pixel’s degree of colour-matching is also affected by those around it: this is analogous to a blur, where lone bright spots will not be as bright after blurring as large clumps of even slightly darker pixels.

To obtain the coordinates of the centre of the blob (allowing the rover to measure the angle between each beacon and thereby determine its position) a weighted average is taken of each pixel according to how well it matches the target colour, after the kernel filter has been applied a specified number of times to the image (labelled as ‘Blobification passes’ in the GUI). This average is then filtered simply by taking the weighted average of the current blob centre (the red crosshair) and the previously calculated blob centre, weighted so that there is only a small contribution from the new centre position. This gives a smoothed position which

will take time to update, but will not be as susceptible to noise if an instantaneous reading is taken.

As seen in figures 16 and 17, this prototype system worked well: it was able to identify blobs of colour even against a noisy background containing other shapes, and although it cannot distinguish blobs by shape or size beyond a certain threshold this should not matter as much when the background is black.

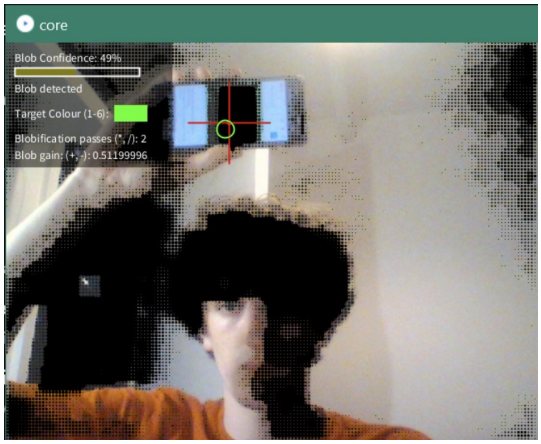

Figure 14: Processing prototype showing kernel filtering approach, detecting a green blob. Matching kernel-filtered pixels are shown as black blobs

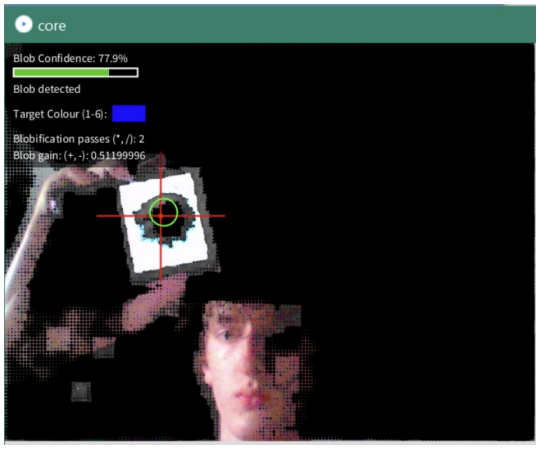

Figure 15: Processing prototype showing kernel filtering approach, detecting a blue blob.

3.4.2 FPGA System

After receiving and setting up the EE2Rover project, there was a period of familiarisation with the SystemVerilog video environment, and of further exploration of the Quartus environment. The RTL FPGA vision system reads one pixel (composed of 24 bits, with 8 bits for each of red, green and blue) from the D8M GPIO camera in every clock cycle, and uses registers to produce information for the whole image: in the case of the EE2Rover sample vision system, it produces a bounding box for matching colours by updating the coordinates of the corners of the bounding box at every pixel. To implement the prototyped vision system, a running total x and y pixel position are kept in registers, to which the current pixel x and y are added respectively if they are 'matching', and at the end of the frame, these totals are divided by the total number of pixels to obtain an average position, stored for transmission, and reset back to 0 for the next frame. The matching is calculated on the basis of the vector distance between the current pixel's R, G and B values and the target's, however since the distance is only compared to a pre-set distance to determine whether it falls within the threshold, an expensive square-root operation is not required.

A user interface overlay was also developed, including an RTL font renderer design, to debug the system and display information on screen, particularly to display what the target colour is in the same colour space as the camera (to visually check how well large parts of the image match), to highlight matching pixels, show both the filtered and unfiltered blob centre crosshairs, and to display when the RTL system receives information from the soft-CPU for debugging purposes. This is shown in figures

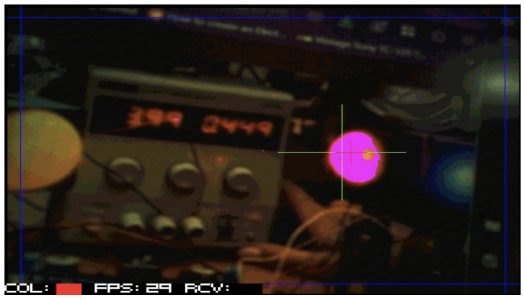

Figure 16: VGA output of beacon detection RTL system searching for red

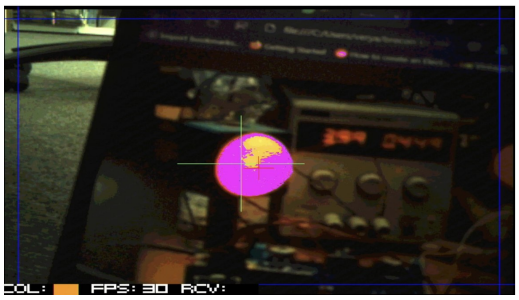

Figure 17: VGA output of beacon detection RTL system searching for orange

The second part of the vision system consists of software running on a NIOS II soft-CPU, onboard the FPGA. This software writes to memory on the RTL system to configure its target colour and tolerance, to search for different beacons, and reads RTL memory to determine the estimated beacon position, and regularly communicates with the central ESP32 over UART, receiving the target colour and transmitting camera information including the target position. The soft-CPU also communicates over the MIPI protocol with the D8M camera to configure its gain, exposure, zoom and focus.

Testing of the vision system was carried out incrementally over time: validating each successive level of functionality. For example, the colour matching system was tested through the feature where pixels deemed to be sufficiently "matching" are replaced with bright purple pixels, allowing tuning of this stage, and once the pixel averaging and blob centre detection logic was implemented, multiple cursors were used to show different metrics of the image such as the single most matching pixel, or average matching pixel position.

3.5 Energy

3.5.1 Overview

The energy system part of the project is intended to harvest solar energy using solar panels to illuminate red, blue and yellow LED beacons which are to be used to locate the position of the rover within the maze through computer vision, as explained previously. The set-up of the energy system is in the form of a DC grid as shown in Figure 18.

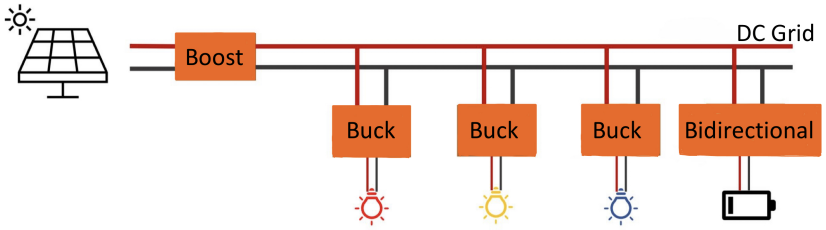

Figure 18: DC Grid Layout

The main energy supply of the DC grid is provided by the PV panels. This energy is injected into the grid using a Switch-Mode Power Supply(SMPS) device operating in open-loop Boost mode. The main task of this SMPS is to implement a Maximum Power Point Tracking(MPPT) algorithm. The energy demand of the system comes from the LED beacons. The energy is extracted from the grid using a LED driver, which acts as a SMPS in closed-loop Buck mode. These SMPSs' main task is to control the current flow going into the LED drivers, which are current-driven. Finally, the energy storage for the DC grid is a supercapacitor, connected to the grid via a bidirectional SMPS, whose main task is also to control the current flow into and

out of the supercapacitor, allowing the charging and discharging of the supercapacitor.

The grid components work together to maintain a nominal grid voltage of about 9-10V. This value was chosen as it is far away from any harsh limits: the lower limit is 7V, set by the minimum input voltage required by the LED drivers. If this limit is reached the beacons will not light up. The upper limit is 16V, set to be far away from the supercapacitor voltage limit, 18V, which, if reached, could permanently damage the grid components.

As all of the SMPSs are connected to the grid through capacitors, the grid itself behaves as a capacitor from the addition of all the individual capacitances. This means that the voltage of the grid can be modeled equation 1. As the grid's objective is to balance the supply and demand of the components, this is the base equation that is used to calculate the amount of current drawn by the beacons depending on the surplus or shortage of voltage in the DC grid compared to the nominal voltage chosen.

$$V_{grid} = \frac{1}{C} \int I_{grid} dt \tag{1}$$

3.5.2 PV Panels

In the design process the initial step was to characterise the solar panels. This was done for both individual PV panels and parallel connections of the three PV panels available. Figures 19 and 20 show the I-V and P-V curves (respectively) for a single PV panel connected across different loads, all rated at 5W. As seen from the figures, the higher the load connected at the output of the SMPS used to characterise the panels, the higher the power drawn from them. Consequently, the characterisation of the parallel connection of the panels was done for the 120 ohm load, as well as the testing in the lab. Figures 21 and 22 show the I-V and P-V characterisation for the parallel arrangement. The parallel arrangement was chosen, as it allows for a higher current output, necessary to cover the current demand from the components in the DC grid.

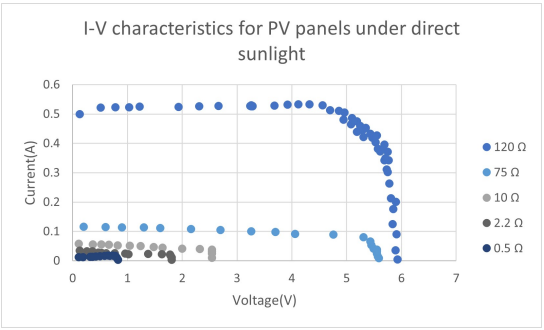

Figure 19: I-V characteristic for single PV panels under direct sunlight

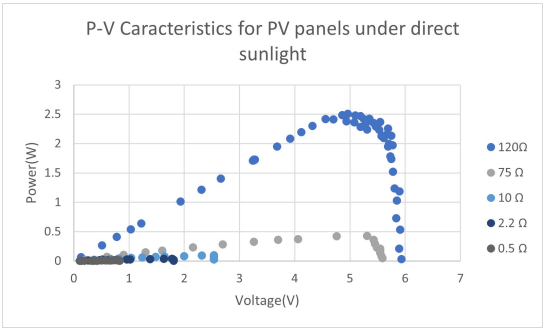

Figure 20: P-V characteristic for single PV panels under direct sunlight

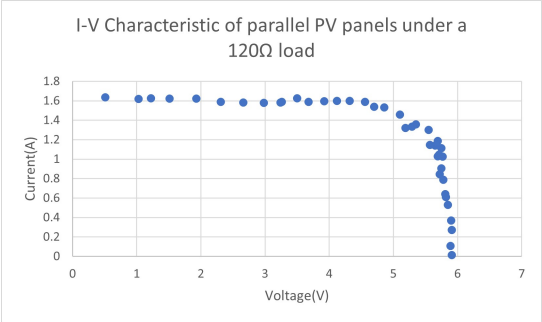

Figure 21: I-V characteristic for parallel PV panels under direct sunlight

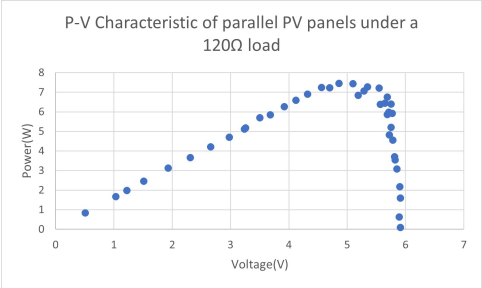

Figure 22: P-V characteristic for parallel PV panels under direct sunlight

Following the characterisation of the PV panels, these were emulated in the lab using a PSU, using a series 148 ohm resistor and a parallel 5 ohm resistor. These resistors were used to better emulate the slopes of the I-V characteristic in Figure inside the lab, as resistors have linear characteristics. The series resistor represents the gentler slope of the I-V characteristic and the parallel resistor the steeper slope.

3.5.3 Maximum Power Point Tracking Algorithm

In order to be able to inject the maximum power into the DC grid from the emulated PV panels, the Boost SMPS sets the duty cycle required to do so. This is done using an MPPT algorithm [7], [8]. Initially, two different algorithms were considered: the perturb and observe(P&O) algorithm or the incremental conductance method. The P&O was chosen due to its higher simplicity due to less complex calculations and its faster response to rapidly changing conditions, as the incremental conductance method could exhibit higher sensitivity to noise.

The P&O algorithm is shown in Figure 23. The duty cycle is shifted towards the MPP by considering the power and voltage changes between two adjacent samples [9]. Although this algorithm correctly tracked the MPP, a lot of oscillation was observed around it, something that would lead to decreased efficiency due to the constant and large adjustments that had to take place. Also, this led to voltage instability at the output of the Boost SMPS, which would disrupt the performance of the rest of the grid elements. This was removed by adding an extra layer of complexity into the code, which set the amount the duty cycle was perturbed by. It was determined that a lot of the oscillation occurred due to noise within the circuit, so it was decided that if the algorithm detected a small power difference between the adjacent samples, the duty cycle would not be perturbed. On the other hand, for greater power differences, the duty cycle would be perturbed in accordance with the P&O algorithm. The increment value was set to 0.004, equal to 1/255, chosen after tuning as it is the step change allowed by the arduino code. The threshold power difference for which the duty cycle was not perturbed was set to be 0.1W, determined from tuning the algorithm. With this change, good MPPT tracking as observed, with little to no oscillation around the MPP.

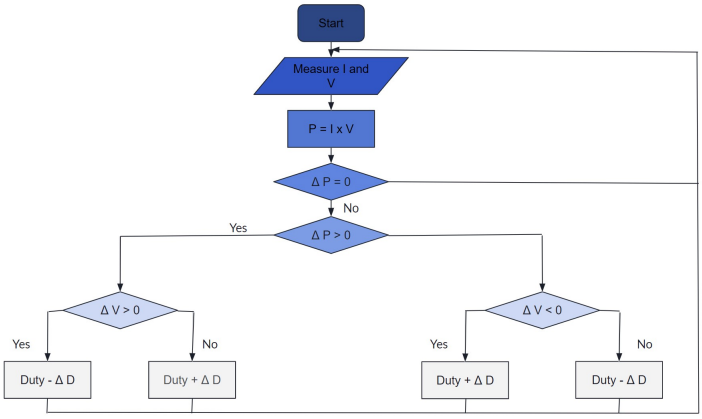

Figure 23: MPPT Flowchart

When testing the Boost SMPS connected at the input to a power supply of 5V and 300mA, and at the output to a load of 120 ohms, the duty cycle needed to output the maximum power was calculated to be 0.627. See Appendix C for the full calculations. When testing the MPPT, it set the duty-cycle to a range of 0.60 to 0.63, showing the design has an average efficiency of about 92%.

3.5.4 DC Grid Control

The next phase in designing the control of the DC grid was considering the different cases for the operation of the DC grid. Four extreme cases were identified: firstly, too much energy being supplied from the PV panels but the supercapacitor is fully charged. The second case is when not enough energy is supplied from the PV panels to operate the LEDs and the supercapacitor is not charged. The third extreme case consists of the DC grid voltage being over the 16V upper limit and in turn, the fourth case consists of the DC grid voltage being under the 8V lower limit. For normal operation, if there was more energy supplied by the PV panels than the LED beacons could consume, the supercapacitor would charge and store the excess energy. On the other hand, if not enough power was supplied to the LED beacons for their normal operation but the supercapacitor was charged, it would discharge and provide the grid with the energy necessary to close the gap between the energy needed by the beacons and the energy supplied by the panels.

Case 1 & Case 2

The first extreme case was solved by drawing more current from the LED drivers. As the LEDs are current driven devices, their brightness is a function of the current through them. Droop control was utilized to keep the DC grid line at the nominal voltage of 9 - 10V, by mapping grid voltage to LED brightness by increasing and decreasing current drawage according to the surplus and lack of voltage in comparison with the nominal voltage of the line.. Therefore, a surplus of voltage is considered anything above 9V, and a lack of anything below 9V.

After characterizing the beacons from the PSU, it was observed that 60mA delivers sufficient brightness, and hence this was set to be the current drawage of the beacons when the line is at its nominal voltage. Current goes down to 25mA for the lowest voltage, and up to 90mA to ensure alignment with the equipment safety limits.

Case 3 & Case 4

As previously discussed, the grid voltage should lie between 7 - 16 V, 7V as to not damage the drivers and 16V from the supercapacitor. To ensure the grid voltage remained within these limits, the drivers were programmed to shut off for an input voltage below 7V, and the Boost SMPS was coded to shut off for an output voltage of over 16V.

3.5.5 LEDs

As previously discussed the LEDs are current driven devices, who will light up for a voltage input of 2V for the blue beacon, and 3V for red and yellow with brightness varying for the current through them. As to ensure good operation of the DC Grid balance between demand and supply must be obtained, this was implemented through droop control [10], [11] in the driver code, by reading supply in the grid line as input voltage and effectively setting demand in the form of current drawn by the beacons.

3.5.6 LED Drivers

Upon testing of the whole system, it was observed that the grid line was successfully lying between the intended 7 - 16V range. The range was split up in steps to deliver current proportionally to the surplus or lack of voltage compared to the 9 - 10V value, whilst keeping in mind equipment safety limits and supply limitations. For the upper range of 12 - 16V, 80mA of current is drawn for each beacon. This value was chosen to ensure that the output of the boost will be sufficient to provide the current needed, whilst keeping in mind testing results that delivered that anything over around 80mA could even be characterized as "wasteful" in terms of brightness. For 10 - 12 V, we are drawing 70mA, which proportionally positions the drawing of60mA for the nominal range of 9 - 10V, further decreasing to 50mA for 9 - 7V. Although product specifications dictate that the drivers will shut off for a voltage input below 7V, it was observed that this was much lower at around 4.5V for this experimental setup. Given that we do not want to harm the equipment, and risk any sort of damage, but we do want to maximize the range used after thorough testing results the decision was taken to still operate them at 6V. The value was chosen, as at 5V there were never problems at all during testing but 2V down from the suggested value, seemed too much of a hazard risk for operational safety. To ensure that operation from 6 to 7V would not produce any unforeseeable complications, current drawn within that range is very small at 25mA. This process can be summarised as a state diagram, shown in figure 24.

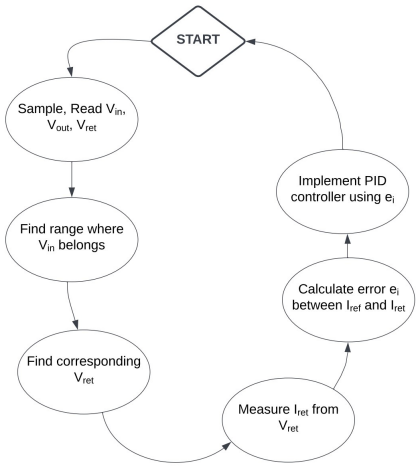

Figure 24: Current Controller State Diagram

The controller used in the LED drivers is shown in figure 25. Current drawn by the beacons is set by a PID current controller whose model follows the one used in the Power Labs. The reference current Iref is set by the input voltage as discussed above and is recreated by calculating the error between Iref and measured value Imeas, which is the input to the controller. In turn, the controller goes on to output the duty cycle needed for the driver to output Iref. This control scheme proved good operation as output current seems to follow correctly reference current for the corresponding input voltage to the drivers.

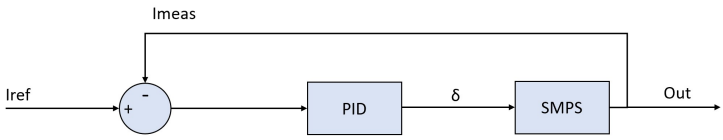

Figure 25: LED Driver Current Controller

To ensure accuracy in our measurements we chose to calibrate the pins of the Raspberry Pi boards, by taking measurements for Vin, Vout, and Vret using the oscilloscope, the known values of input voltage from the PSU, as well as the resistor value R = 1.02Ω, that output current flows out from.

To ensure that at 6V the drivers cease to operate, we set reference current to 0, whilst implementing current limit using the current sensing resistor to prevent the beacons from being damaged. Given that the resistor is about 1Ω, the voltage across it is almost equivalent to the current through the beacons and hence for Vret $>$ 0.35 V the duty cycle value is reduced, which in turn reduces output current.

3.5.7 Supercapacitor

The bidirectional SMPS that connected the supercapacitor to the grid did not arrive on time for its integration in the energy system. However, testing was carried out with the supercapacitor connected directly to the grid. For no power out from the Boost SMPS, the supercapacitor's compensation was sufficient to keep the beacons on for around 10 seconds. The disadvantage of this setup compared to the connection through the bidirectional SMPS, is that the capacitor charges and discharges for any change in voltage of the grid line, thus releasing and drawing energy aimlessly at times.

3.5.8 Integration of DC Grid

Upon integration of all components the challenge seemed to present itself in the early stages of coding the drivers. MPPT and Drivers had both been tested from the PSU at the point, but not together.

The beacons were constantly illuminated when they were connected to the PSU, but when they were powered from the boost they were flashing for the lower range of input voltages. Upon testing the problem was located at Vret, which dropped below 110mV when flashing began. At that point the driver code consisted of a PID controller with one Iref value of 60mA for the whole range of input voltages. The problem was fixed by introducing two reference currents one at 60mA, and one at 40mA, by taking a measurement of Vret and using the first one when it was above 110mV, and the second for lesser values.

However, this proved to be counter-intuitive as reference current was effectively set by output current, changing the focus from trying to keep the LEDs at constant brightness to letting them reflect the voltage of the grid, leading to the final pre-discussed design.

A model that set reference current Iref directly from the difference between Vin and 9V was attempted but proved to be unsuccessful due to a lot of oscillation, and difficulties in tuning, leading to the final model which covers the grid's and the beacons' needs more than comfortably.

3.6 Design

3.6.1 Initial Ideas

When first constructing the robot, we used the vertical design which was provided to us. We decided on this over the horizontal counterpart because of the advantages provided for a self-balancing design. The first design for the rover is shown in figure 26.

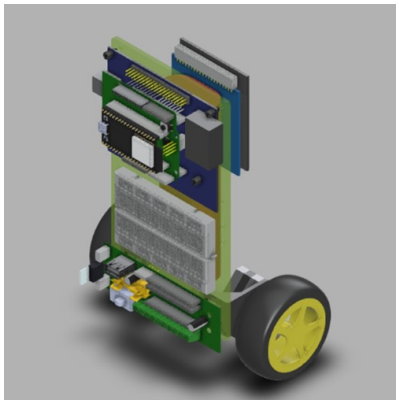

Figure 26: Original Robot Vertical Design

The vertical distribution of mass makes it much easier to evenly distribute the mass of the components on either side of the wheels, compared to the horizontal design which has a long platform stretching out far from the wheel base. We also chose the vertical design since we found it was the more aesthetically pleasing of the two.
After successfully balancing the above robot, it became apparent there were many issues to be resolved. Firstly, the single-sided nature of the design meant that the equilibrium position of the robot when balancing was offset from vertical, we found this made it more difficult to manoeuvre the robot, as well as angling the camera down towards the ground which was very bad for beacon recognition.
Additionally, there was also a need for more space to mount an additional ESP32 and circuitry, along with somewhere to mount LDRs for detecting the walls of the maze.

3.6.2 First Design Iteration

Our first design iteration effectively took the existing vertical design and made it double-sided, greatly increasing the space available for mounting components while also allowing us to achieve an equilibrium position much closer to vertical. This was possible because the battery could be placed in between the two sides, therefore having the majority of its mass directly above the wheel base. Furthermore, we were able to place other heavier components, such as the FPGA, lower down, shifting the centre of mass closer the ground and improving balancing performance. The front of this design is shown in figure 27 and the back in figure 28.

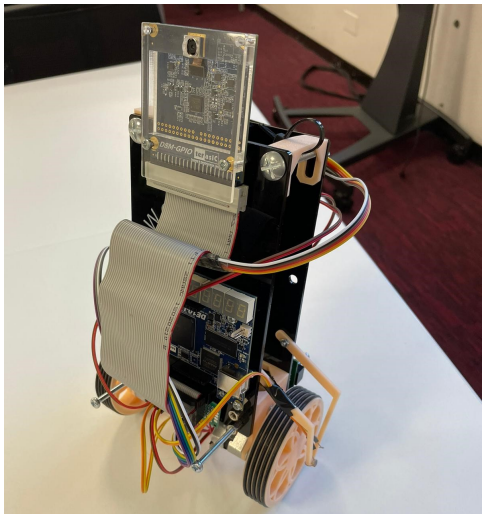

Figure 27: Completed vertical 'sandwich' design - front

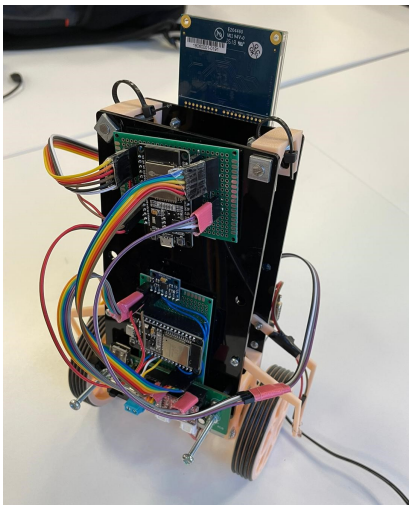

Figure 28: Completed vertical 'sandwich' design - back

Apart from creating the two laser cut sides for the robot, a few other parts had to be designed and 3D printed before the robot could be assembled. The first of these was a dual-sided bracket for attaching the motors and wheels. To start with, we took the existing motor bracket and mirrored the design across the wheel axle. This worked fine but after building the rover we felt the two sides were too far apart, so a second bracket was constructed which placed the motors further below the two sides, allowing for a much slimmer design. The redesigned bracket is shown in figure 29. A top brace for supporting the structure was also created which can be seen in figure 30.

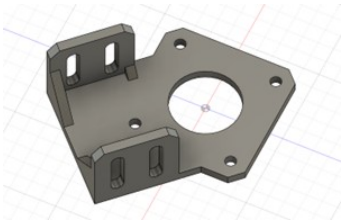

Figure 29: Redesigned motor bracket

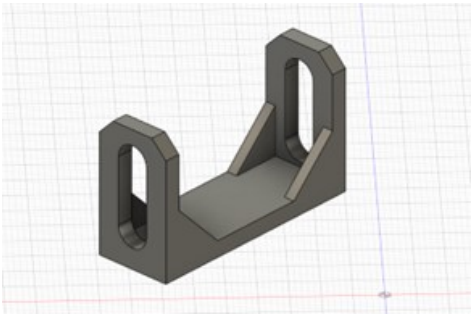

Figure 30: Model of top brace

While creating the first design we also decided to improve the wheels. The uneven tread pattern on the provided wheels caused some problems when balancing as the rover would tend to roll into the grooves in the tread rather than being free to rest at any point on the tyre. Additionally, creating larger wheels would lead to more friction between the tyres and the surface, further improving balancing performance.

We designed and printed larger wheels with an 80mm diameter, at first leaving the rims flat and using electrical tape as tyres. These are shown in figure 31. This did not prove as effective as hoped, we found that the wheels would occasionally slip causing the robot to sometimes lose control and fall.

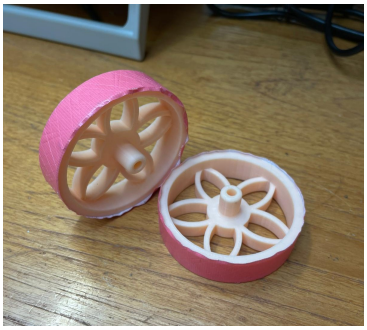

Figure 31: First wheel design

For this reason, we designed a second set of wheels maintaining the 80mm diameter but now utilising O-rings as tyres, shown in figure 32. These wheels performed very well, creating an even, high friction connection between the robot and the surface.

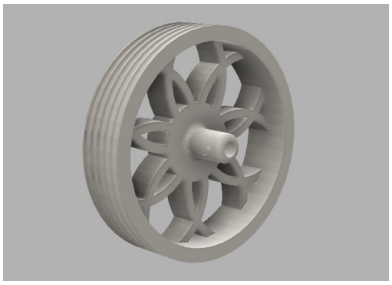

Figure 32: 3D render of the final wheel design. Note the grooves for the O-rings to rest in

The final consideration for the design was to create brackets for the LDRs. The front and back LDRs could simply be attached to the bottom of each laser cut piece, but the side LDRs were more difficult to mount. Brackets were designed which suspended the LDRs over the wheel axle so that they would allow for consistent readings by always maintaining the same elevation while the robot changed in pitch during motion and general balancing. These are shown in figure 33.

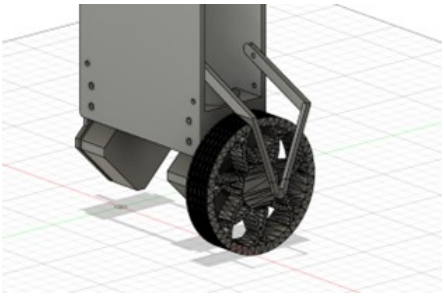

Figure 33: Model showing LDR bracket position over wheel

3.6.3 Second Design Iteration

After testing our robot in the maze, we found that although it could balance and manoeuvre successfully, this was only possible at very low speeds. This was because the pitch could only be varied by about two degrees before it was impossible to correct and prevent the robot from falling.
For this reason, we developed a new design which was much smaller than before, while still maintaining

many of the original design components.

As seen in figure 37, the new design consists of two horizontal layers stacked rather than two vertical pieces. In this way it was possible to massively decrease the height of the robot, leading to a much lower centre of mass than before.

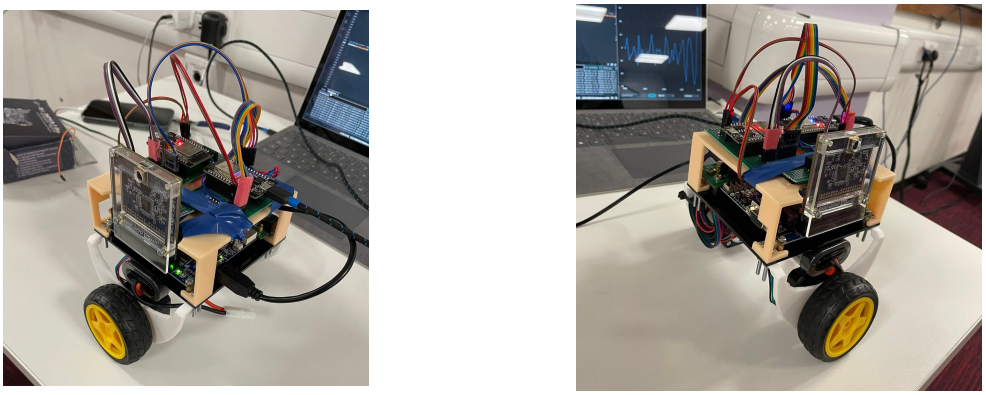

Figure 34: New Design

Other features worth noting are the new brackets for the motors which encompass the battery, again permitting a very short design. Also, the camera is now facing sideways so that the FPGA can be oriented as shown, further decreasing the overall size of the design.

The redesign proved very successful, considerably improving the balancing and manoeuvrability of the robot. This allowed for faster motion and subsequently a significant improvement in the time to solve the maze.

4 Evaluation

4.1 Results

4.1.1 Control

For results of the control system see the balancing video in Appendix F.

4.1.2 Networking

The results from the front-end and database have been promising, providing a quick and intuitive showcase of how the rover operates, as well as quick data processing for the rover to do its tasks efficiently.

4.1.3 Beacon-based location

Evaluation of position estimation through manual measurement was determined to be inaccurate and excessively time consuming, so an automatic evaluation method was devised: a simulation was run where many rover positions and the beacon angles they would result in were simulated, and, with a Normally distributed error to angle measurements added, position estimations obtained through the same method used by the rover system. These were then plotted against a number of variables including distance from the maze origin and x- and y-axis positions to investigate any correlations. The angle errors were chosen with Normal standard deviations of 5 degrees which accurately represents typical measurement error for the vision system, and a more optimistic 1 degree error to investigate what might be achieved if the vision system were improved.

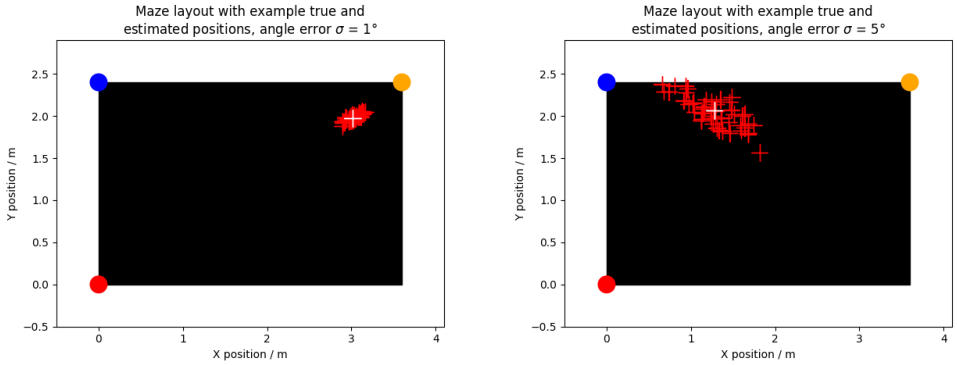

Figure 35: Example of position estimation simulation with 1 and 5 degree simulated error: the 'true' position is in white, from which beacon angles with simulated noise are calculated, with the position estimates determined from these by the Tienstra method of resection shown in red. (the positions are not comparable and only show indicative locations within each simulation, so are only for illustration of the process)

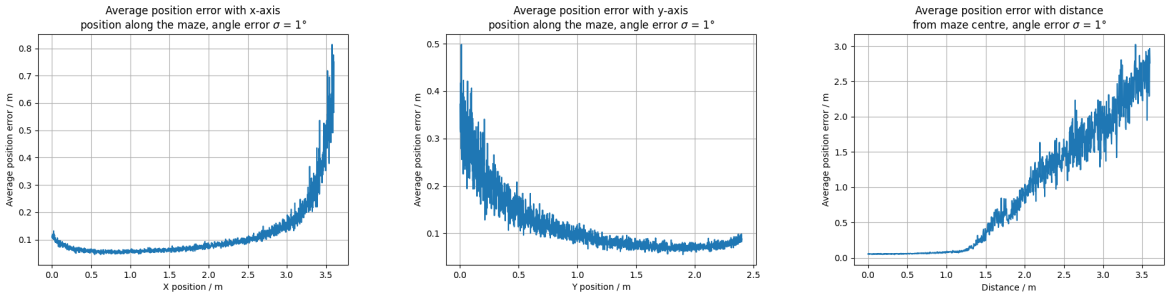

Figure 36: Simulation position errors with respect to x-axis position, y-axis position and distance from the maze centre for a Normally distributed simulated angle error with standard deviation of 1 degree.

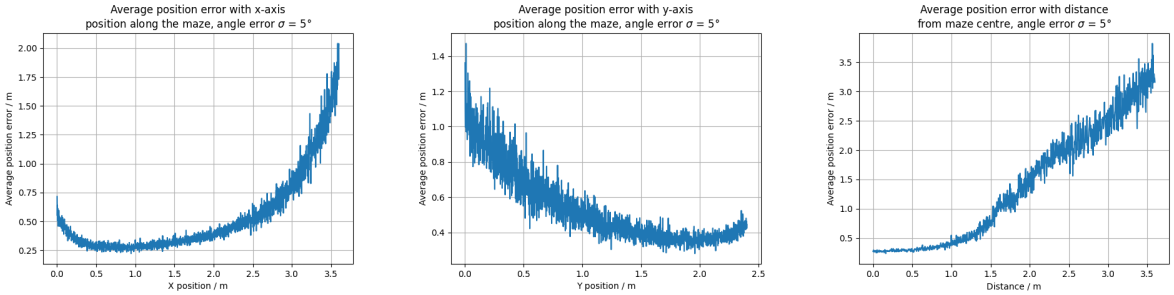

Figure 37: Simulation position errors as in previous figure, with simulated error standard deviation of 5 degree.

These results show that any deviation from the centre of the maze rapidly results in increased errors, up to the order of over a metre for 5 degree angle measurement error. They suggest that the rover design, in its determination of current position by combining beacon and inertial sources of information, should rely less on beacon positioning as it reaches the extremities of the maze, but that sufficiently close to the centre beacon measurement can provide a good source of position recalibration. They also show that although clearly the 1 degree error results in better position estimation, the error would still be unacceptable at the edges of the maze, so a position fusion approach is still necessary.

4.1.4 Energy

Testing Process: Testing of the current controller consisted of powering the beacons directly from the PSU to determine if they draw the correct current. Vret was measured from the lower output pin (Vout) of the beacons, which in turn was multiplied by 1.02 to determine Iret.

After testing the current controller directly from the PSU to confirm correct current draw, the beacons were connected to the boost smps to confirm whether the grid was succesfully kept at 9V.

Results: As observed Iret follows correctly the set Iref values from the code for each range of input voltage from the grid, with small variations that do not present a problem limiting current to the correct range.

For an input voltage from the PSU down to 3.77V the grid line was successfully kept at 9V at all times. For lower input voltages, the grid line value dropped, which was to be expected as the current needs of the beacons could not be satisfied from the boost output.

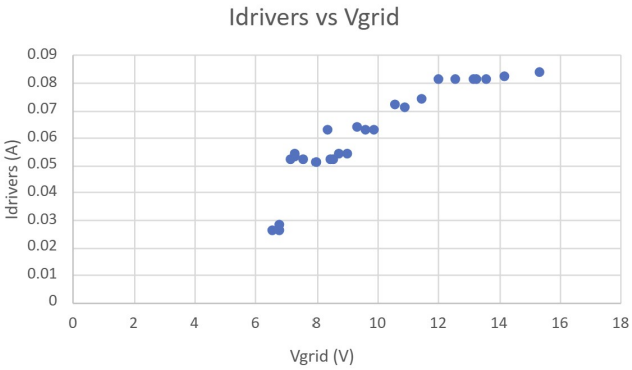

Figure 38: Current controller results

To observe how the DC Grid works as a whole for input voltage ranges from 0 to 5V see Appendix F for a video demonstration.

4.2 Project Cost

The bill of materials for each part of the project are shown below. This covers all the materials used that were not provided by the department as part of the project, given to each group in a box at the beginning of the project. The costs do not include the control system and the communication system as the components used were provided in the box or do not contribute to the costs as they only use software.

Table 1: Bill of Materials for Command System

Component	Quantity	Price (inc VAT)
Perfboard	1	£0.33
Header block	5	£0.10*5 = £0.50
Header wire	25	£0.05*25 = £1.25
Total		£2.08

Table 2: Bill of Materials for Energy System

Component	Quantity	Price (inc VAT)
IRF530NPbF MOSFET	1	£0.23
Total		£0.23

Table 3: Bill of Materials for Rover Structure

Component	Quantity	Price (inc VAT)
O-rings	2	£3.25*2 = £6.50
Black acrylic		£9.93
Total		£16.43

Table 4: Project Costs

Component	Price (inc VAT)
Command system	£2.08
Energy System	£0.23
Rover structure	£16.43
Total	£18.74

The total cost for this project is £18.74, which is well under the budget assigned. The prices for any circuitry components take into account the components having been bought in bulk, meaning that if only the number of components needed were bought, the price would have been elevated by an average of 24%(assuming the number of components bought in bulk were 50+), leaving the price at £23.24.

The prices of the components obtained from the Imperial EEE Department have been taken as being those listed in the ordering app. Also, the prices incude the VAT, currently at 20% in the UK. If removed, the production cost of the rover would be lower.

In addition, only direct costs have been considered in this analysis. Further costs that would have to be included in the production of such a product, such as administrative costs, energy costs, etc. During the duration of this project, these have been covered by the Imperial EEE Department.

4.3 Improvements

The team was content with the results that were obtained. Due to the constraints present, both in timing and resources, there would have been some improvements that could have been made. This section discusses these improvements.

4.3.1 Control

The control system can be enhanced through data collection for a more accurate model, code optimization using call-backs, implementation of a sophisticated controller through pole placement, and more dedicated tuning time. These improvements will improve system performance, efficiency, and precision, meeting the desired operational objectives.

4.3.2 Networking

One improvement that could be added on is the use of error checking on the server side, with database data or past readings to check if the position received from the Esp and Camera make sense, and as such being able to correct these errors where present.

The front-end design also leaves a lot to be added, while it is still functional, it is very simple and could use some additional react components to enhance the clarity of the rovers movement or allow greater interaction with the user.

4.3.3 Command

The command and control system serves its purpose well: one of the main advantages is its extreme simplicity and robustness, and it is hard to envisage many alterations to this subsystem specifically which would actually improve the rover: one potential option is the use of more sophisticated and robust communication protocols: currently none of the protocols used are inherently reliable, and use no validation/checksum beyond parity bits present in the physical protocol. The risk of data corruption in the system currently is low as any erroneous values received would very quickly be corrected by subsequent packets, but with more development it would be possible to switch over to more complex and reliable protocols, potentially

improving system reliability and performance.

4.3.4 Vision

In future, if more time were spent improving the design, the main limitation to be overcome would be the single-colour matching nature of the design. This approach is simple to test and iterate on, and allows very simple integration with the rest of the rover, however it does not permit truly simultaneous detection of multiple beacons at once, potentially reducing location performance and limiting the max rotation rate in the scanning state. Allowing the detection of multiple colours leverages the ability of the RTL system to perform many computationally expensive operations simultaneously with no loss of speed, removing the current bottleneck imposed by requiring the command ESP to regularly switch colours, and wait for the moving average to settle before detecting the next one.

The other major area of improvement to investigate would be the use of kernel filtering: although it was decided not to develop this for the rover it would theoretically allow greater noise rejection and stability, and would be particularly useful for hypothetical further iterations of the rover operating in more challenging environments.

4.3.5 Energy

Even though the SMPS for the supercapacitor did not arrive, the code for it was developed in Arduino.

As stated previously, the DC grid acts as a capacitor due to the individual capacitances in each component. From equation 2, it can be observed that if $\frac{dV_{grid}}{dt}$ is equal to 0, all the currents in the components making up the DC grid are balanced.

$$\frac{dV_{grid}}{dt} = \frac{1}{C}(i_{boost} - i_{LED} \pm i_{capacitor}) \tag{2}$$

This SMPS would be coded to allow positive current flow into the supercapacitor in order to charge it and negative current flow out of the supercapacitor to discharge it. The charging would take place when the power provided by the PV panels was larger than the power demanded from the LEDs and the supercapacitor was not fully charged. Essentially, this means that in this case, $i_{boost} > i_{LED}$. On the other hand, discharging would take place when there was not enough power provided by the PV panels to drive the LEDs and the supercapacitor was not empty, or $i_{boost} < i_{led}$. In summary, the function of the bidirectional SMPS is to use current control to set the duty cycle of the SMPS in order to bridge the gap between supply and demand in the dc grid.

The current control of the bidirectional SMPS was designed to follow the controller in figure 39. The outer loop of the controller generates the reference current Iref. Firstly, the current grid voltage would be read by the SMPS and compared to the desired nominal grid voltage (9V) to give dV. Then the error would be passed through a P controller, which sets the reference current to be proportional to the error, as well as positive or negative depending on the capacitor having to charge or discharge. A P controller was used instead of a PID controller as dV might never be equal to 0, meaning a PID controller might integrate the error to infinity, which would lead to unstable system behavior.

The inner loop of the controller is used to then control the SMPS current. The current through the bidirectional SMPS would be measured and the current error between the desired current and the actual current would be passed through a PID controller to set the duty cycle of the SMPS to the value needed to obtain Iref.

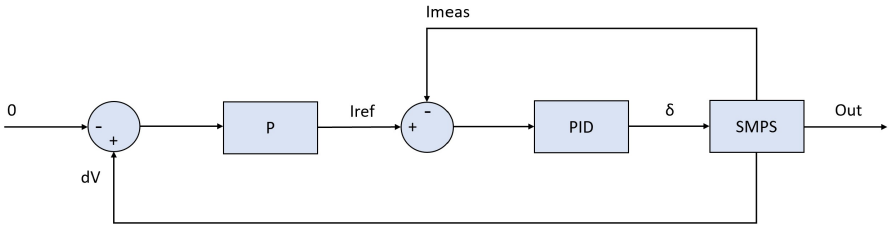

Figure 39: Supercapacitor Controller

In order to test this controller, it was coded in the SMPS board used for the MPPT algorithm. Using this board meant that the voltages used to perform the tests had to be considerably lower than the voltages that would be used using the actual bidirectional SMPS board meant for this part of the project. The device was tested using a bidirectional power supply in the set-up shown in figure 40. The PSU was set to 0.76 A as the current limit for the capacitor was 0.38A. To test positive current flow(charging), the voltage was set to 10V(which is above 9V), and R was set to 26.3Ω. To test negative current flow(discharging), the voltage was set to 5V(below 9V) and R to 13.2Ω to achieve the correct current division. It achieved a bidirectional current, showing it worked as intended but to conclude if it would actually work with the intended board at the correct voltages more thorough testing would have to take place.

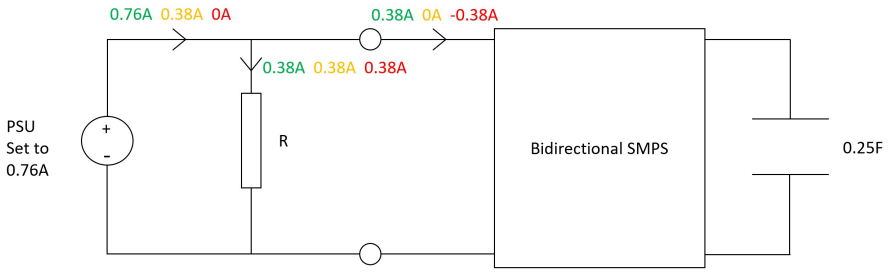

Figure 40: Testing set-up

Moreover, another way of implementing current control on the supercapacitor so that it would charge when the grid voltage was above 9V and discharging it when the grid was below 9V was researched and designed. This method consists of using a MOSFET as a bidirectional current device by making use of its body diode [12].

This could be achieved by connecting the supercapacitor across the MOSFET so that when it was off, the current would flow from the source to the drain through the body diode, charging the supercapacitor. The supercapacitor would then be discharged when the MOSFET was on, as the current would flow through the MOSFET channel from drain to source.

The MOSFET chosen was the IRF30NPbF. This choice was made as it is a large channel MOSFET, meaning that the drain-to-source on-resistance is low($90m\Omega$), leading to less power loss, which is essential for a DC grid. As this MOSFET is an NMOS, the following equation must be satisfied to turn it on : $V_{GS} > V_{TH}$ and $V_{DS} > V_{GS} - V_{TH}$ [13]. As $2V < V_{TH} < 4V$, by applying 3.3V at the gate of the transistor from an Arduino 3.3V pin, it could be turned on. In order to actually implement this, the grid voltage would have to be read by the Arduino, and if the voltage was below the nominal voltage, meaning the supercapacitor needed to discharge, apply the 3.3V at the gate to turn the MOSFET on.

As of the time this report was written, there had been no time to implement or test this but the team decided to try to implement it for the demo day (23/06/2023).

4.3.6 Design

Although having the camera facing sideways enables us to have a very compact robot design, it is possible this could cause difficulties obtaining accurate headings of the beacons since the robot will vary in pitch slightly whilst maintaining its balanced position. Therefore, with more time it would be best to iterate on the design again and if possible, produce a robot just as compact but with the camera facing forwards. This could lead to more consistent beacon heading readings.

It would also be useful to experiment with different wheel sizes because even though larger wheels increase friction with the surface, they also require more torque to be driven and the stepper motors only have so much power available. Therefore, through testing different sizes it would be possible to find the optimal size for our application, resulting in the best overall movement and balancing performance.

5 Conclusion

In this paper we have outlined the design of a rover which successfully tackled the problem of mapping a maze and finding the shortest path through. By making use of an MPPT algorithm, a DC grid was set up that employed almost all available power, successfully balancing power supply and demand through current control, lighting up the beacons even with no power in from the PSU by utilising a supercapacitor. Through design of an RTL system, beacon camera coordinates were successfully obtained and passed via the associated softcore processor over to the command and control ESP32, which in turn allowed the estimation of the rover's position in the maze, making use of Tienstra's method of resection. By taking in sensor readings from the accelerometer and gyroscope, and employing sensor fusion, an angular acceleration was calculated and implemented to stabilise the rover. Upon integration of all discussed parts the final design effectively and efficiently achieved the desired goal.

6 References

[1] "Physics - Test Your Knowledge: Moment of Inertia (16 of 24) Wheel and Spoke: Part 1/5," www.youtube.com. https://www.youtube.com/watch?v=kc$_{A6}$04YvA (accessed May. 23, 2023).

[2]"Understanding Kalman Filters," uk.mathworks.com. https://uk.mathworks.com/videos/series/understanding-kalman-filters.html (accessed May. 26, 2023).

[3] "THREE POINT RESECTION PROBLEM." Available: http://www.mesamike.org/geocache/GC1B0Q9/resection-methods.pdf

[4] "ESP32 SPI API under the Ardino IDE,"Arduino Forum, Jan. 16, 2021. https://forum.arduino.cc/t/esp32-spi-api-under-the-ardino-ide/691603/6 (accessed May. 27, 2023).

[5] "Tienstra formula,"Wikipedia, Mar. 16, 2023. https://en.wikipedia.org/wiki/Tienstra$_{f}ormula$ (accessed May. 29, 2023).

[6] "Processing.org," Processing.org, 2019. Available: https://processing.org/

[7] F. Z. Hamidon, P. D. A. Aziz, and N. H. M. Yunus, "Photovoltaic array modelling with PO MPPT algorithm in MATLAB," 2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE), 2012, Accessed: Jun. 19, 2023. [Online]. Available:https://www.academia.edu/9498743/Photovoltaic$_{A}rray_{M}odelling$

[8] M. Kamran, M. Mudassar, M. R. Fazal, M. U. Asghar, M. Bilal, and R. Asghar, "Implementation of improved Perturb Observe MPPT technique with confined search space for standalone photovoltaic system," Journal of King Saud University - Engineering Sciences, Jun. 2018, doi: https://doi.org/10.1016/j.jksues.2018.04.006.

[9] "LAB # 07 Implementation of Maximum Power Point Tracking (MPPT) Algorithm Using MATLAB and Simulink." Available: https://lms.su.edu.pk/download?filename=1588431464-lab-07-updated.pdflesson=15071

[10] R. Kumar and M. K. Pathak, "Distributed droop control of dc microgrid for improved voltage regulation and current sharing," IET Renewable Power Generation, vol. 14, no. 13, pp. 2499–2506, Oct. 2020, doi: https://doi.org/10.1049/iet-rpg.2019.0983.

[11] A.-C. . Braitor, G. C. Konstantopoulos, and V. Kadirkamanathan, "Current-Limiting Droop Control Design and Stability Analysis for Paralleled Boost Converters in DC Microgrids," IEEE Transactions on Control Systems Technology, vol. 29, no. 1, pp. 385–394, Jan. 2021, doi: https://doi.org/10.1109/tcst.2019.2951092.

[12] "Charging and discharging a capacitor automatically with MOSFET," Arduino Forum, Jun. 14, 2023. Available: https://forum.arduino.cc/t/charging-and-discharging-a-capacitor-automatically-with-mosfet/1137803 (accessed Jun. 19, 2023).

[13] Behzad Razavi, Design of analog CMOS integrated circuits. New York Mcgraw Hill Education, 2017.

7 Appendix

7.1 Appendix A

The following equations outline the mathematical calculations necessary to determine the required equations for the implementation of the Kalman filter:

$$x = \begin{bmatrix} \theta \\ \dot{\theta}_b \end{bmatrix} \quad \hat{x}_{\bar{k}} = A\hat{x}_{k-1} + Bu_k \tag{3}$$

$$\hat{x}_{\bar{k}} = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \hat{x}_{k-1} + \begin{bmatrix} \Delta t \\ 0 \end{bmatrix} u_k \tag{4}$$

$$\hat{x}_{\bar{k}} = \begin{bmatrix} \theta_{k-1} - \Delta t\dot{\theta}_{bk-1} + \Delta t u_k \\ \dot{\theta}_{bk-1} \end{bmatrix} \tag{5}$$

$$\begin{aligned} P_{\bar{k}} &= \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} P_{k-1} \begin{bmatrix} 1 & 0 \\ -\Delta t & 1 \end{bmatrix} + \Delta t Q \\ &= \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\Delta t & 1 \end{bmatrix} + \Delta t \begin{bmatrix} Q_\theta & 0 \\ 0 & Q_{\dot{\theta}b} \end{bmatrix} \\ &= \begin{bmatrix} P_{00} + \Delta t(-P_{10} - P_{01} + \Delta t P_{11}) + Q_\theta & P_{01} - \Delta t P_{11} \\ P_{10} - \Delta t P_{11} & P_{11} + Q_{\dot{\theta}b} \end{bmatrix} \end{aligned} \tag{6}$$

$$K_k = \frac{P_{\bar{k}} C^T}{C P_{\bar{k}} C^T + R} = \frac{\begin{bmatrix} P_{00} \\ P_{01} \end{bmatrix}}{P_{00} + R} \quad where \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{7}$$

$$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(y_k - C\hat{x}_{\bar{k}}) \tag{8}$$

$$\begin{aligned} \hat{x}_k &= \begin{bmatrix} \theta^- \\ \dot{\theta}_b^- \end{bmatrix} + \begin{bmatrix} K_{k0} \\ K_{k1} \end{bmatrix} \begin{bmatrix} y_k - \theta \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \theta^- + K_{k0}(y_k - C\theta) \\ \dot{\theta}_b^- \end{bmatrix} \end{aligned} \tag{9}$$

$$\begin{aligned} P_k &= (I - K_k C)P_k^- \\ &= \begin{bmatrix} P_{00}^- & P_{01}^- \\ P_{10}^- & P_{11}^- \end{bmatrix} - \begin{bmatrix} K_{k0} \\ K_{k1} \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} \end{bmatrix} \\ &= \begin{bmatrix} P_{00}^- & P_{01}^- \\ P_{10}^- & P_{11}^- \end{bmatrix} - \begin{bmatrix} K_{k0}P_{00} & K_{k0}P_{01} \\ K_{k1}P_{00} & K_{k1}P_{01} \end{bmatrix} \end{aligned} \tag{10}$$

A : state transition model
C : observation model
Q : covariance of noise process
R : covariance of observation noise

B : control input model
U_k : control vector
y : observation of state
y_k : pitch from accelometer

7.2 Appendix B

The following equations demonstrate the mathematical conversion process from an analog PID controller to a discrete PID controller using the backward Euler method:

$$\begin{aligned}\bar{R}(z) &= K_p + K_i\frac{T_s z}{z-1} + K_d\frac{z-1}{T_s z} \\ &= K_p + K_i T_s\frac{1}{1-z^{-1}} + \frac{K_d}{T_s}(1-z^{-1}) \\ &= \frac{1}{1-z^{-1}}(K_p(1-z^{-1}) + K_i T_s + \frac{K_d}{T_s}(1-2z^{-1}+z^{-2})) \\ &= \frac{\frac{K_d}{T_s}z^{-2} - (K_p + 2\frac{K_d}{T_s})z^{-1} + (K_p + K_i T_s + \frac{K_d}{T_s})}{1-z^{-1}}\end{aligned} \tag{11}$$

7.3 Appendix C

Here is the derivation of the ideal duty cycle for the MPPT for Vin = 5V, Iin = 300mA, connected across a load $R_L = 120\Omega$.

$$Pin = 5 * 0.3 = 1.5W \tag{12}$$

$$Pout = \frac{Vout^2}{R_L} \tag{13}$$

For Pout to equal Pin, Vout needs to be 13.42V. The voltage ratio for a Boost SMPS operating in continious conduction mode is shown in equation 14.

$$\frac{Vout}{Vin} = \frac{1}{1-\delta} \tag{14}$$

Solving, $\delta = 0.627$.

7.4 Appendix D

Here are adjoined the datasheets of the components used:

MOSFET IRF530NPbF
`IRF530NPbF.pmd(infineon.com)`

7.5 Appendix E

Here is the PDS of the project:

- Life in service: the rover should be able to perform and solve the maze for as long as the battery supplies an adequate voltage. Following this, the battery will need to be replaced with one that is fully charged. The other components should be able to perform successfully for a very long time.
- Maintenance: regular maintenance not required, design robust. All fasteners will be standard and most cables should be easy to replace with standard header wires. Perfboard design will be modular so components are easily replaceable.

- Competition: not applicable.
- Shipping: not applicable.
- Packing: not applicable.
- Quantity: only one rover has been produced.
- Manufacturing facilities: not applicable.
- Size: rover needs to be small enough to fit down the narrowest corridors of the maze. The dimensions are: length 19.3cm, width 12cm, and height 12.7cm.
- Weight: the final weight is 0.9555kg.
- Aesthetics, Appearance and Finish: design conveys that it is meant to balance on two wheels only. Unnecessary clutter of cables etc. has been cleaned up where possible.
- Materials: design is produced with black acrylic, PLA and the required electronic components.
- Product life span: not applicable as it is a product that will not go into production.
- Standards and specifications: the rover would be designed according to European Standards.
- Ergonomics: as the rover is autonomous, the user would not have to have any previous knowledge.
- Quality and reliability: the rover has been built to be reliable and robust.
- Shelf life: NiMH battery will slowly degrade over time. Will need to be charged or even primed if not used for a long time.
- Processes: parts have been manufactured using 3D printing and laser cutting.
- Time-scale: the time scale for this project was 5 weeks.
- Testing: the testing took place continuously throughout the project of the individual components as well as testing during one week to integrate all the individual parts.
- Safety: constraints have been put in place inside the code controlling SMPSs in the grid in order for voltage limits not to be reached so electrical components aren't damaged or explode. If any wires loose their insulation these will need to be changed.
- Company constraints: the constraints are discussed in 1.2 - Project Design Specification.
- Market constraints: this type of product is very niche so the market is very small.
- Patents, literature and product data: the rover is not patented at the moment.
- Political and social implications: not applicable.
- Legal: not applicable.
- Installation: not applicable.
- Documentation: not applicable.
- Disposal: after service the rover should be taken apart to recycle components and dispose safely of the batteries.

7.6 Appendix F

Here are adjoined the links to videos of each of the components working:

The YouTube link where all of them can be found is:
https://www.youtube.com/@WOBBL-E

The control system:
https://www.youtube.com/watch?v=UHQGRH455Lw

The vision system:
Beacon recognition:
https://www.youtube.com/watch?v=fZmAKt-2U4Y
Vision and turning test:
https://www.youtube.com/watch?v=6p2-guWUJuA

The energy system:
https://www.youtube.com/watch?v=me93$_A pU1kw$