Imperial College London

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

ELEC50008: ENGINEERING DESIGN PROJECT

Group 7 (CPB2): Mars Rover Project Report

Authors:

Jubo Xu
CID: 01874153
Jx1820@ic.ac.uk

Justin Huang
CID:01860968
Jh4420@ic.ac.uk

Charlie Chen
CID: 01848992
cc1420@ic.ac.uk

Jacky Jiang
CID: 01860908
jj1220@ic.ac.uk

David Zheng
CID: 01861514
dz820@ic.ac.uk

Ziduan Li
CID: 01864493
zl3820@ic.ac.uk

Rui Li
CID: 01856441
rl1020@ic.ac.uk

Submission date: June 22, 2022
Submitted to: Dr. Bouchaala, Adam

Contents

1. Introduction

This project aims to design and construct a Mars rover for extra-terrestrial exploration. The rover system must be able to navigate itself in the test arena and build a map indicating the locations of aliens and their underground power infrastructure. The rover system must also autonomously avoid the alien and the infrastructure while running. In addition, the real-time system status should be displayed on the web application. The web application should have the capability of controlling the Rover remotely. Furthermore, there must be a system for powering the Rover from solar energy for long-term sustainable operation on Mars.

2. Design Hierarchy

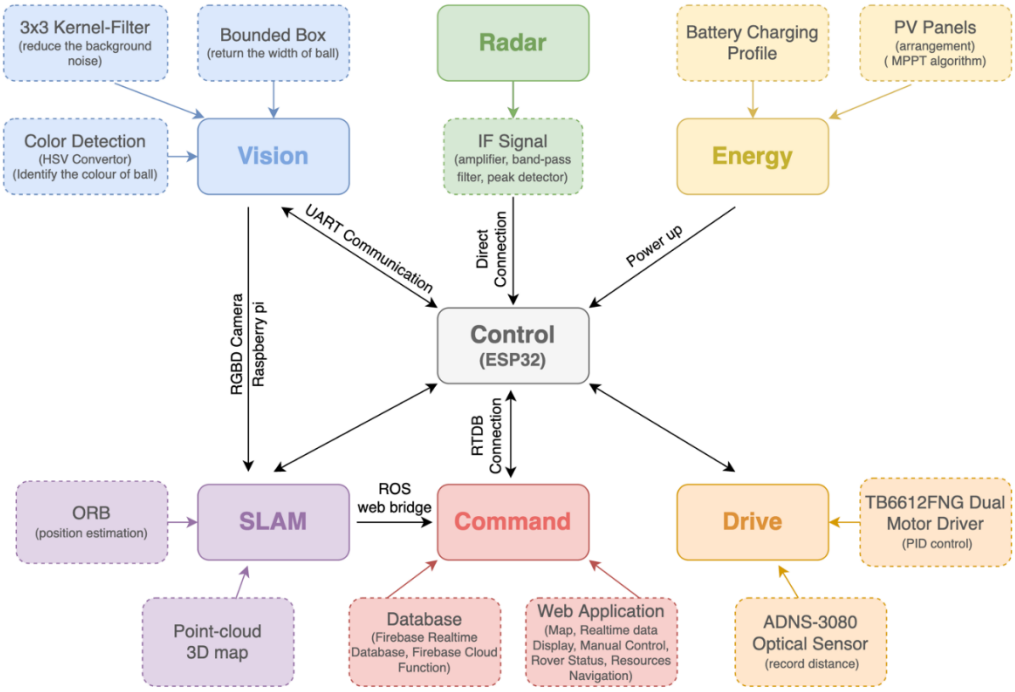


Fig 1: Design Hierarchy

The rover system consists of 6 modules: Control, Vision, Radar, Energy and Drive.

The control module is based on a Espressif 32-bit microcontroller. The eps32 provides on-board processing abilities and interfaces between submodules. The Vision module identifies and locates the alien and their infrastructure in the field of view using the Terasic D8M-GPIO camera and the FPGA. The FPGA communicates with control module via UART protocol. The controller reads serial data sequentially from the FPGA and updates the characteristics of aliens and the position of their infrastructure. Upon receipt of these information, the control module command the drive module for rover maneuver and obstacle avoidance. The radar module detects the underground alien infrastructure using the HB100. The module is directly connected to the analogue pin of esp32. When the signal level is above the threshold value, the controller records the current coordinates for further processing. The energy module provides the system with charged power bank using solar energy. The energy module should guarantee optimal power yield from the solar panel and high-power efficiency. Furthermore, the ESP32 has Wi-Fi functionality, making wireless communication between the rover and web server possible. The communication is based on the RTDB connection. The control module uploads the rover status and current position to database and receives commands from web applications for manual rover control. The highlight of the design is the use of Simultaneous localization and mapping (SLAM). The video is recorded by computed on Raspberry Pi 4B to generate point-cloud 3D map.

3. Rover submodules

3.1 Energy

3.1.1 PV panels arrangement

A photovoltaic system was highly susceptible to partial shading [1], which could lead to severe power loss. Partial shading can also cause local maxima in P-V characteristics that is not expected as local maxima can create errors in MPPT. Reducing the effect of partial shading is therefore crucial to the energy system design. The PV panel given consists of 10 cells in series, each provides a voltage of around 0.5V and 5V in combination[1]. To test the susceptibility, two cells were shaded completely. For a single panel, when two cells were covered, the current through all 10 cells was hugely reduced. As a result, the power generated was reduced from 0.23W to 0.0065W, a 97.2% decrease in its performance. Local maxima also occurred in the P-V curve shown in Fig 2, also in [1].

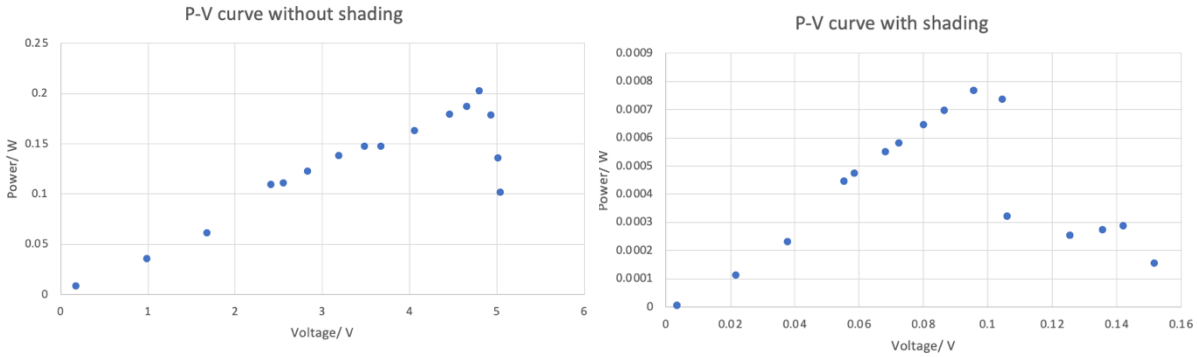


Fig 2: P-V curves with and without partial-shading

To mitigate the effect of partial shading, the method of four-panel-in-parallel was adopted, giving out optimal power of 4.6W, voltage of 5V and current of 0.92A. As four panels were connected in parallel, the system's voltage was slightly affected if there was one panel shaded, and there was no need to consider local maxima anymore. Under the same light condition as above, power generated now reduced from 0.23W to 0.2047W, representing a degradation of performance of only 11% this time compared to 97.2%.

3.1.2 Grid interface

A grid interface was needed to perform multiple tasks to achieve maximum power point tracking of the PV panel and to control the voltage between 4.5V & 5.2V according to the power bank's rating.

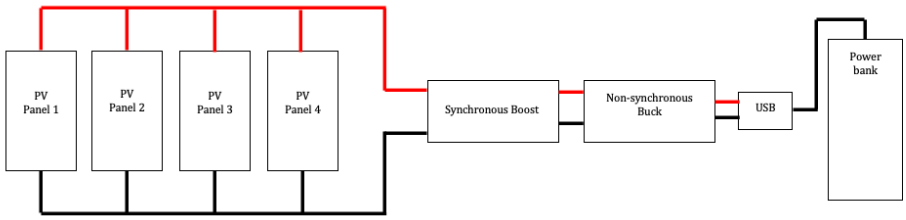


Fig 3: The layout of the power station

In order to perform these tasks, two SMPSes were used, as one SMPS could only output one PWM. As shown in Fig 3, the first SMPS was used to achieve MPPT using a synchronous boost, and the second SMPS was used to achieve voltage regulation using a non-synchronous buck.

[1] Schematics available in Appendices 8.1

3.1.3 MPPT Algorithm

Maximum power translation is always important in power system. The maximum power of the solar energy system should always be harvested. Therefore, it is crucial to characterise the power-voltage behaviour of the Photovoltaic system.

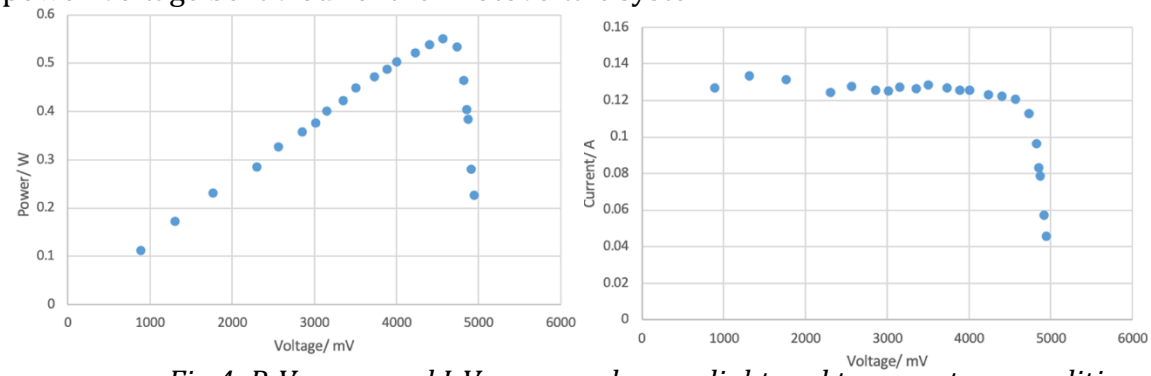


Fig 4: P-V curve and I-V curve under one light and temperature condition

In order to track the maximum power, perturb-and-observe method [2] is considered first. By using a Boost SMPS, output voltage was perturbed, if output voltage was perturbed to the right and the power after perturbation was higher, meaning the maximum power point was still to the right, then PWM was further increased by 0.01, and vice versa.

Nevertheless, there were two drawbacks [2] when using perturb-and-observe. First, the MPP was never accurately located as the voltage was constantly perturbed to find the closest maximum power point. Second, the irradiance of the light changed severely in between two sampling processes of the perturb-and-observe algorithm which had compromised the effort of the algorithm.

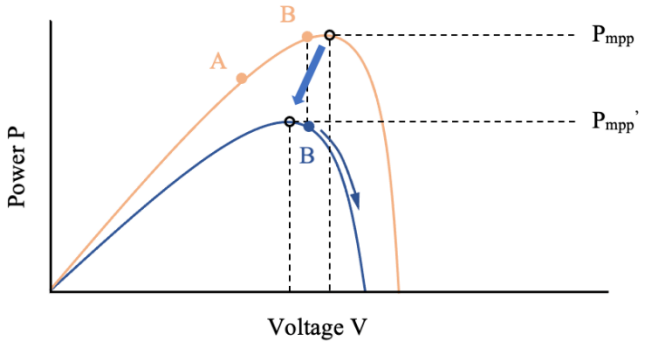


Fig 5: Algorithm fails to find MPPT

For example, in Fig 5, point A and B represent the two sampling processes of the algorithm, both laying on the left-hand side of the MPP. The P-V curve changed and now B lays on the right of the MPP, but the algorithm still went as before, and the power was moved further away from the MPP.

A second algorithm, incremental conductance [2], was then considered. By inspecting the P-V curve, the MPP is the only point that has zero gradient, where $\frac{dP}{dV} = 0$. As $P = I \times V$, $\frac{dP}{dV} = \frac{d(I \times V)}{dV} = I + \frac{V \times dI}{dV}$ and if the sampling process is small enough dI, dV can be approximated to $\Delta I, \Delta V$. It turns out that at maximum power point, $\frac{\Delta I}{\Delta V} = -\frac{I}{V}$, where $\frac{\Delta I}{\Delta V}$ is called incremental conductance and $\frac{I}{V}$ is the instantaneous conductance. If $\frac{\Delta I}{\Delta V} < -\frac{I}{V}$, the MPP is to the left and the PWM is reduced by 0.01; if $\frac{\Delta I}{\Delta V} > -\frac{I}{V}$, the MPP is to the right and the PWM is increased by 0.01. By implementing this method, the MPP can be detected more accurately as it does not hover

around the MPP like the perturb and the observe algorithm (proven by the detection process of maximum power point shown in serial monitor of Arduino).

3.1.4 Battery charging

In order to design the second SMPS, the I-V characteristic of the battery has to be determined first. The power source is used to provide the voltage, which has to be strictly limited between 4.5V and 5.2V to prevent the battery from damage.

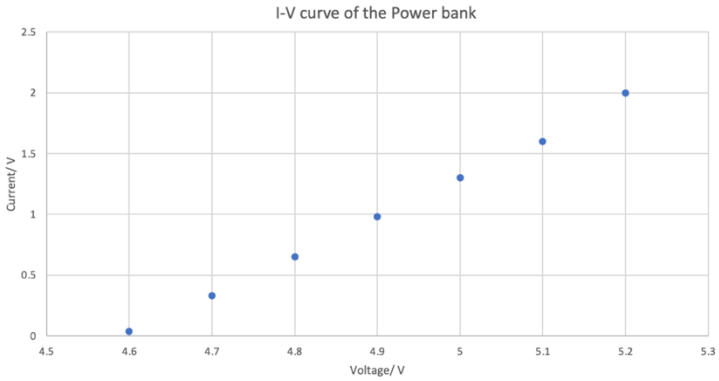


Fig 6: I-V curve of the battery

Arduino can be coded that, if output voltage is within the safe range, current PWM is output; if output voltage is < 4.5V, PWM increase 0.01; if output voltage > 5.2V, PWM decrease 0.01. The output voltage is then controlled in the operating region.

3.2 Drive

3.2.1 Design

The Drive Module receives instructions from the Control Module. The Drive Module executes the instruction (go straight or go rotation). The rover will either travel in a straight line for a fixed distance or rotate for a fixed angle.

3.2.2 Optical Sensor

The optical flow sensor measures the distance travelled in x and y direction with respect to the rover frame of reference. The data type is changed to float and sampling frequency is set to 100 Hz. A 5V LED light strip is attached at the rear of the rover to improve lighting and surface quality.

3.2.3 PID Control

Due to the difference in friction and motor output power, the rover cannot travel in a straight line. PID control algorithm is introduced to make sure the rover follows a straight line.

The delta duty cycle is proportional to the current error, previous error, and the difference between current and previous error. The integral error of the controller and the delta duty cycle are bounded to avoid overshoot. In addition, a two-motor straight line control strategy is employed. After tunning and testing, the proportional coefficient is set to be 0.015 and the integral coefficient is set to be 0.01.

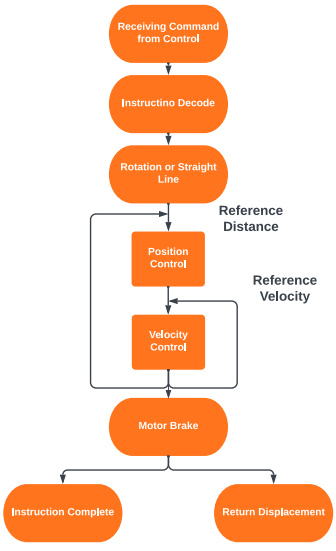


Fig 7: Drive Module Flowchart

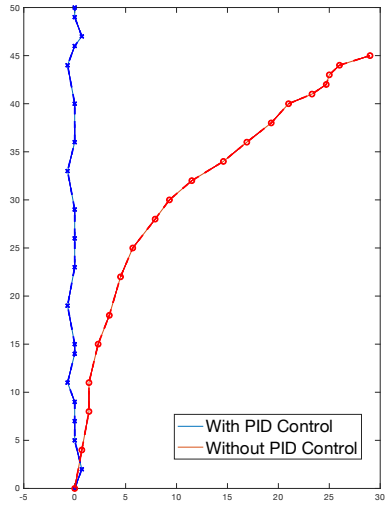


Fig 8: PID Control Testing

3.2.4 Position and Angle Control

The rover position and angle control are implemented using a dual loop. The inner loop controls the rotational speed of the two motors. The outer loop keeps tracks of the current position and position setpoint. For rotation control, the total distance traveled in tangential direction equals to the arc length described by the rover's optical sensor. The arc length is the product of angle setpoint in radian and radius of rotation (the distance between the optical sensor and the midpoint of the rover shaft). Considering that the two motors do not output the same power, we also add a proportional controller to the angle control to maintain the center of rotation. The total distance traveled in radial direction should be controlled to zero.

3.3 Command

3.3.1 Design

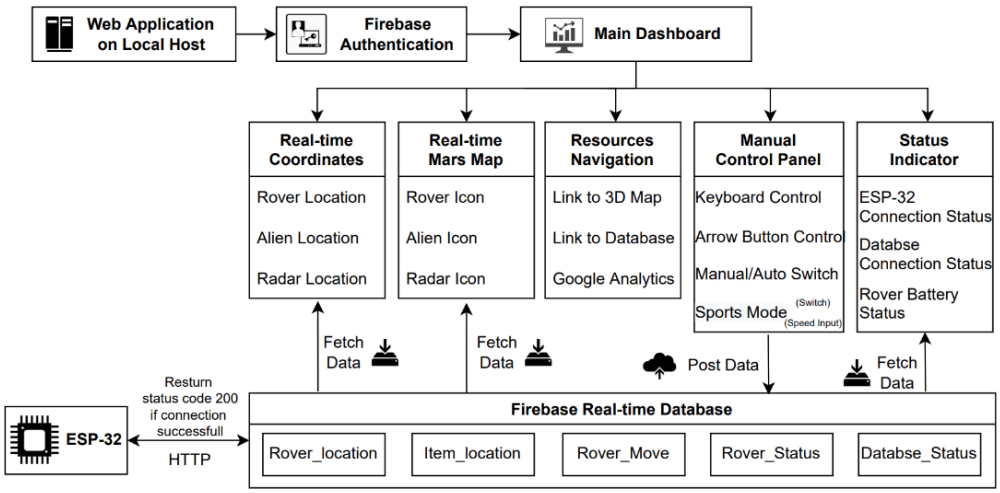


Fig 9: Overview of Command subsystem

3.3.2 Database

Instead of traditional SQL database, we used a cloud-based NoSQL database, Firebase Realtime Database, which offers the following merits that matches our request. [10]

-Accessibility:

Make the database accessible remotely is one of the crucial functionalities for testing and real-life application. Thus, we choose cloud database, enabling us to control the rover and display its info from/to anywhere of the world. Among them, firebase real-time database be accessed directly from a web browser, so an application server might not be necessary, making it even more accessible. (Although due to other considerations, we still use a serverless firebase endpoints, which will be discussed in backend)

-Real-time synchronisation:

The real-time feature of the firebase database will enable the data transmit between website and esp-32(discussed later in the Control Module) seamless with no noticeable delay. Thus, web app can respond immediately to changes.

-Data Structure & Visualisation[1]:

Firebase Realtime Database store data in JSON tree as nodes instead of tables and records, which enable horizontally scaling of data that SQL is perfect for handling large data sets as data logging are happening in real time. (Will be discussed later in the Control Module) Although SQL supports more programming languages, and more powerful in complex manipulation of data, these features do not serve a meaningful purpose in this project.

Thanks to the Firebase inbuilt online console, we can visualise the real-time data traffic, logs, detailed user information, and errors supported by Google Analytics[2].[12]

3.3.3 Frontend[2]

The frontend consists of a landing page with authentication and a command interface.

-Security:

Security is crucial to prevent malicious activities in the command system. An authentication page includes login form authenticating the user for email & password was integrated with web app. Firebase Authentication provides inbuilt table that stores user credentials saving time to manually set up extra table.

Fig 10: Authentication

-Efficiency:

We use TypeScript which provides more robust tooling such as error detection before run-time and type inference which helped save time in debugging. We include an open-source Material-UI which provides comprehensive range of styled UI tools and components that can simply integrate with React, thus maintained time-efficiency without compromise of functionality and style.

-Modularity & Concision:

Building upon the React framework, the command interface composes of 5 functional components in grid: Rover Status, Realtime Data Display, Map, Manual Control and Resource Navigation. Components are written individually which is in accordance with the high-level modularity design. Separating components in grids also offers opportunities for future feature updates without having conflicts between components which may potentially cause redesign and reorganise of the structure of the interface. Finally, components were integrated as a single compact command interface which enables intuitive user interactions with without routing to different pages.

3.3.4 Backend

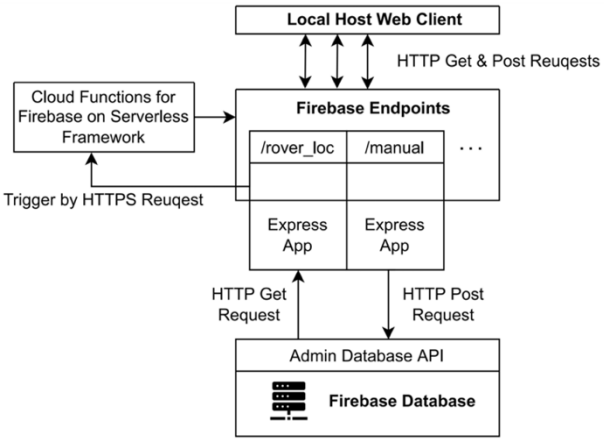


Fig 11: Overview of Backend Infrastructure

Backend server post/fetch data between database and web, based on NodeJS environment instead of python as we consider the speed (multithreading) , universality between front & backend, the rich npm library is more important. [5][13]

[2] Appendix 8.3.1 Screenshot for Web Page[1] Appendix 8.3.2 for JSON Tree Data Structure Example

[7] Appendix 8.3.3, 8.3.4 for Screenshot

-Connect Web to Database:

It is connected through firebase endpoints on a serverless framework as shown above, which make our data accessible remotely without the need of dedicated server. We use endpoints instead of directly access the database as there's trade-off between the use slightly more resources with significant time need to re-develop our frontend code after switching the database. We also tried to access the database directly from web clients using firestore database with its encapsulated native API, however, it is difficult to connect the firestore database which featuring mobile app development to ESP-32.[8][9]

-Suitable Database API:

Admin Database API is chosen to access the real-time firebase database, which allow us to save to or retrieve from the JSON tree in database by initialise with full or limit server privileges then reference to the desired branch of data.[6][11]

We choose directly apply admin API without rest API because the following reasons:

- Rest API enable complex data queries and analysis, but increased complexity with higher latency. However, we have no data processing on command.
- Without rest API, NodeJS client will maintain internal version of any active data, in case of disconnection, it will be synchronised later when connected. Enabling off-line write to database without rest API can improve the reliability and robustness.[4]

3.4 Radar

3.4.1 Design

A HB100 radar is used to detect the underground alien infrastructure, which are fans with metal covered on their blades. The radar emits a signal at the frequency of 10.5GHz. When it is transmitted to a moving object, there will be a frequency difference Δf between the transmitted and received signal. That is output as the IF signal of a HB100 radar. Since Δf is proportional to the object's moving velocity, we can detect a target moving at a constant velocity by observing whether an expected Δf is detected. The target - fan rotates at a velocity of 10m/s and results in an IF signal of 366Hz.

We design a circuit that filters out all undesired signals, only allowing signals at the target frequency (366Hz) pass through and connect the IF signal to it. The circuit will have an output only when the target is detected by the radar. A peak detector is added before transmitting the output to the ESP32 to give it a DC analogue signal. Once ESP32 detects a signal, it means the target is in the radar's range. Then, the rover will stop and rotate to detect the direction with the strongest signal.

3.4.2 Amplifier Circuit

We obtained the largest and the clearest IF signal when the target is 2cm away from radar. However, it's only 22mV. The amplitude is very small and contains a lot of noise.

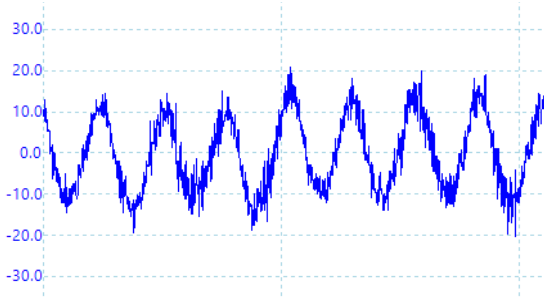


Fig 12: original IF signal at distance of 2cm

Based on the site measurement, the vertical distance between the radar and the target is about 8cm when radar is right above the fan. To detect the signal at a further distance, the original IF signal need to be amplified first. The amplifier circuit we used is the one provided in the application note of HB100 [14]. It's also called Limpkin's amplifier circuit [14]. The measured bode plot of the circuit shows that the signal can be amplified by a gain of 42dB. The high frequency noises have all been removed since the circuit consists of two band-pass

filters with low cut-off frequencies. Now, a clear signal at target frequency can be detected at up to 10cm. However, a 50Hz low-frequency noise occurs when the target distance is greater than 5cm. That might because the amplification of the target signal isn't large enough compared with the noise. Overall, we still need another circuit to filter out all potential undesired signal.

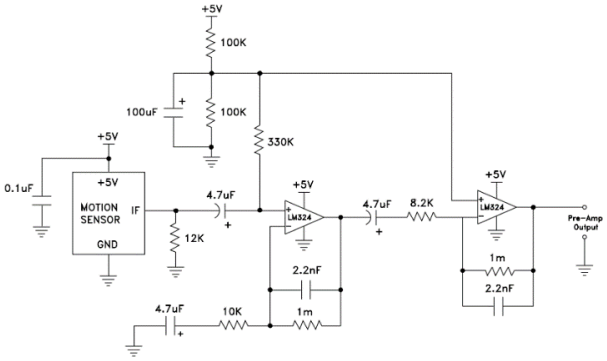


Fig 13: amplifier circuit

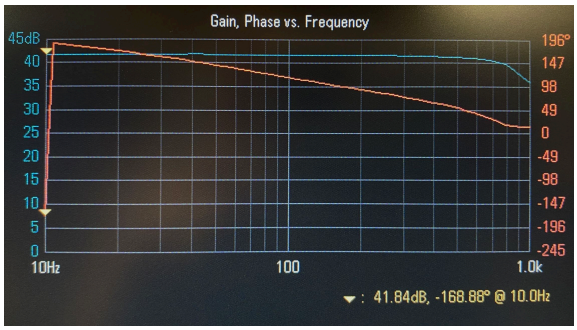
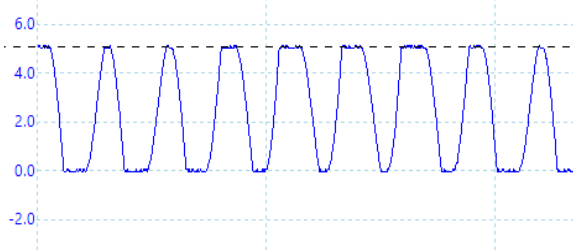


Fig 14: measured bode plot



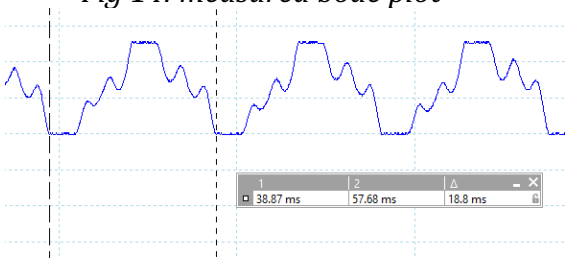


Fig 15 & 16: output at distance of 2cm and 9cm

3.4.3 Band-pass Filter

A band-pass filter with center frequency of 366Hz is needed to get a more precise target signal. Here we use a two-stage band-pass filter with a pass band of around 300Hz and center frequency at 366Hz. Considering of the variation in blades' rotating velocity which leads to change in Δf under different power supply, the filter has a relatively large pass band. The bode plot shown as below and the 50Hz is successfully removed.

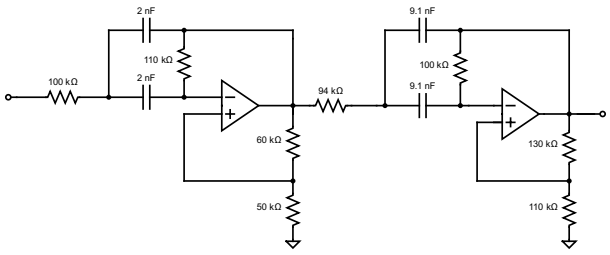
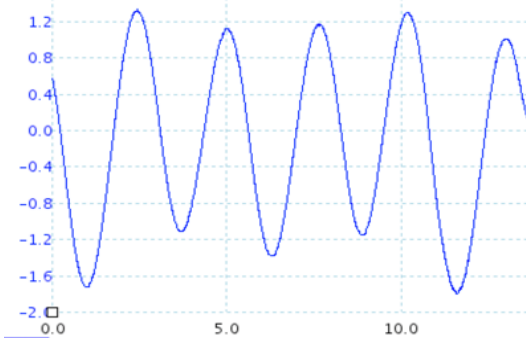


Fig 17: band-pass filter circuit

Fig 18: measured bode plot



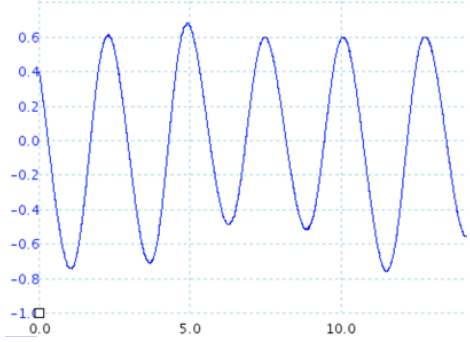


Fig 19 & 20: output at distance of 9cm and 12cm

3.4.4 Target Location Identification

The output now is large and precise enough for us to detect the target. The next step is to input it into ESP32. ESP32 can read an analog signal and determine the voltage value. To simplify the determination, a peak detector is used before that to convert the harmonic signal to a DC voltage. The final output is shown on the right-hand side.

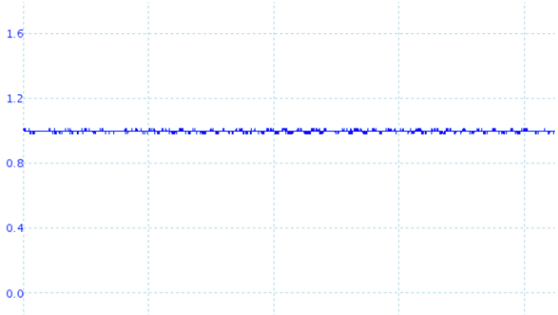


Fig 21: output after peak detector at 8cm

When testing, the target and the rover were kept relatively static. After thorough data analysis, we observed that the detected signal strength varies with changes in rover rotation angle and this signal is at its peak level when the rover is facing the target (0 bearing with respect to the line that the target is located). Therefore, the location identification logic is as follows:

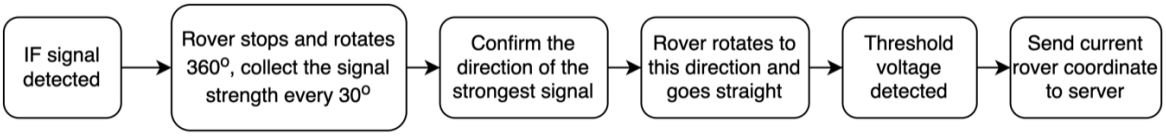


Fig 22: location identification logic

The detailed values of threshold voltage and the signal strength at different angles can be seen in appendix[3].

3.5 Vision

3.5.1 HSV Convertor and Colour Detection

RGB is the colour space we are most exposed to, red (R), green (G), and blue (B). HSV is to use hue H, saturation S, luminance V to describe the change of colour. HSV is better suited for image processing and RGB is better suited for colour rendering presentations. [3]

$$h = \begin{cases} 0^\circ & \text{if } max = min \\ 60^\circ \times \frac{g-b}{max-min} + 0^\circ, & \text{if } max = r \text{ and } g \geq b \\ 60^\circ \times \frac{g-b}{max-min} + 360^\circ, & \text{if } max = r \text{ and } g < b \\ 60^\circ \times \frac{b-r}{max-min} + 120^\circ, & \text{if } max = g \\ 60^\circ \times \frac{r-g}{max-min} + 240^\circ, & \text{if } max = b \end{cases}$$

$$s = \begin{cases} 0, & \text{if } max = 0 \\ \frac{max-min}{max} = 1 - \frac{min}{max}, & \text{otherwise} \end{cases}$$

$$v = max$$

Fig 23: RGB to HSV Formula

3.5.2 3x3 Kernel-Filter

After transferring the RGB data into HSV form, there are lots of background noises, and most of these noises are single pixels, so a filter is needed to filter out the noise. Our filtering is a kind of erosion process, which means a colored pixel will be maintained if most of its surroundings are in the same color. To implement this kind of filter, a 3x3 filtering kernel is needed to do the convolution. To generate the 3x3 kernel, shift-register (RAM-based) is used. The assignment and connection between incoming pixel, shift ram, and output kernel is shown in the figure. Each row of this matrix behaves as a kind of 3 words shift register. The principle of the kernel generation is shown in the following figures, where the red pixel is the incoming pixel. This design makes sure that when one pixel comes in, 9 elements of the matrix will come out immediately, which will avoid the time synchronization issue of valid and ready signal. The problem with such a design is that the convolution is not complete or not valid for the first two columns of the whole image, but since image dimension is 480x640, side effects can be ignored.

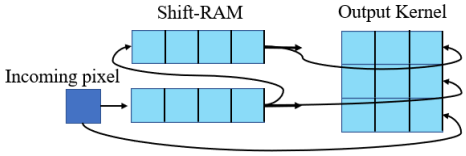


Figure 24: Connections

[3] Appendix 8.4

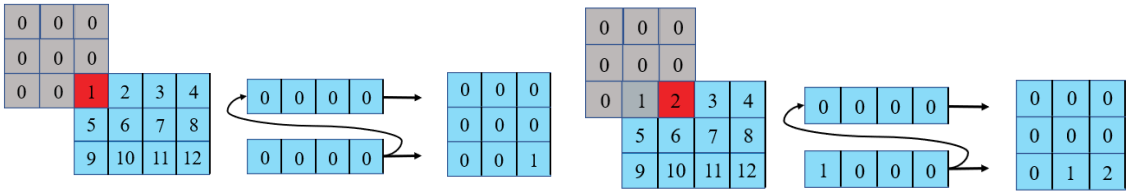


Figure 25: Process at cycle 1 and cycle 2

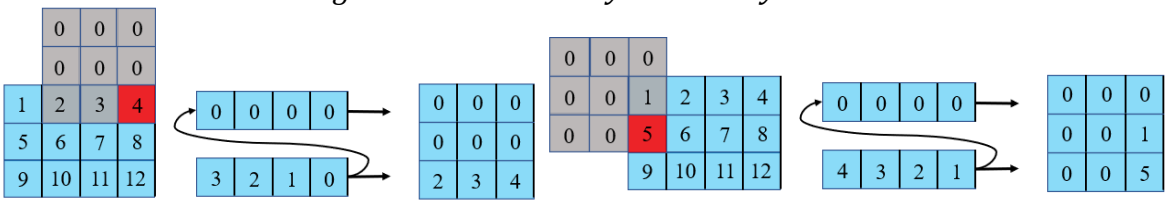


Figure 26: Process at cycle 4 and cycle 5

The logic of the filter is simple: if all the 9 elements are equal to a specific color, then the convolved output pixel will also have this color, if any element is not, then the output is not. The problem of this filter is that although all the single pixel noise can be filtered, the detected target will also have an erosion, to compensate for it, we could add a filter after it, which will set the output to be in this color if any element of this kernel is in this color.

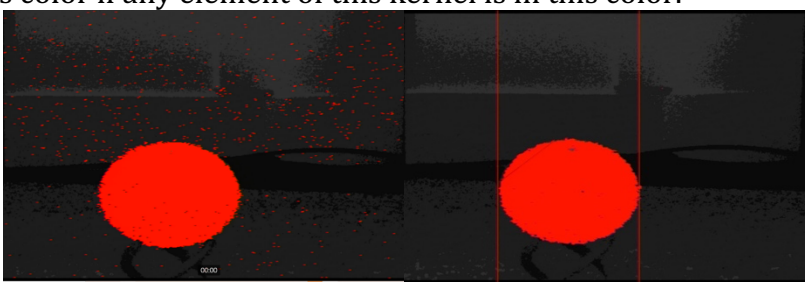


Fig 27: Before and After Filtering

3.5.3 Building Detection

To detect the building, we first increase the contrast of the grey image and add a threshold to separate the black, white, and other grey level. The SOBEL filtering is then implemented to detect the edges of between the black stripe and the white stripe.

Our emphasise algorithm is a kind of localised greyscale stretch method. In each kernel, we find the maximum and minimum grey values of 9 elements, and the output value will be equal to $255 \times (input\ value - minimum)/(maximum - minimum)$, which expand the small greyscale interval into 0 to 255 and improve the contrast.

The threshold implementation is quite simple, since after the emphasise the value of black and white stripes are quite near to 8'd0 and 8'd255, so our upper threshold is 240, which means when the value of the pixel is larger than 8'd240 (near to white) the pixel will be set to white; our lower threshold is 20, which means if the pixel is very near to black then it will be set to black. For other pixels, they will be set to 8'd100, which is light grey.

3.5.4 Bounding Box

After noise filtering, we need to calculate the pixel width of the ball and mark the ball so that we can calculate its center point and pixel width by locating the left and right boundaries of the ball. In order to reduce the impact of rest noises on the detection of the pellets, first, we adjusted the pixels with Y coordinate in the range of 0 to 200 on the screen to grey to eliminate the noise. Then, we set up 10 shift registers to store the previous pixels, when they have the same value, we can consider them as part of the aliens. Then, updating the most left and right boundaries by comparing x values of registers and recent boundaries.

3.5.5 UART Communication between FPGA and ESP32

UART can send the data we have processed in the vision module to the ESP32 sequentially over a reliable wired connection. The FPGA can send 32-bit binary encoding at a time and the ESP32 can receive data sequentially at 8 bits at a time. In order to improve the efficiency of the use of bits, we represent the deviation of the center point position and the distance from the ball to the rover in every 8 bits.

The bit [3:0] is used to represent 15 levels from 20 cm (level 1) to 90 cm (level 15), with each level representing 5 cm, if the ball locates outside of this range, the distance level is 0. The bit [5:4] indicates the deviation of the center point, 0 indicates the center point of the ball is on the left side of the screen and 1 indicates the position on the right side of the screen. Only when it is equal to 2 can it be determined that the ball is in the center of the screen and that the distance detected at this time is effectively tangential.

3.5.6 Simultaneous localization and mapping (SLAM)

The optical sensor on the rover is not very accurate, so there would be accumulated error in our coordinate calculation. Visual SLAM is a good way to help us estimate the position and gesture of the camera and generate the map, especially the 3D cloud-point map. Visual SLAM has four stages, the first one is visual odometry, which is used to estimate the displacement and rotation of the rover by finding distinct or important elements in the image and calculating their displacement in pixel coordinate between several frames. However, there are noises of the image and Accumulating Drift due to the accumulated error. We need to add Backend-Optimization and Loop Back Detection. Backend-Optimization is mainly used to estimate the state of the entire system from the noisy data and calculate the Maximum-a-Posteriori of the estimation. The Loop Back Detection is to determine whether the camera is coming back to its original position by finding the similarities between some frames.

Our SLAM is based on an opensource SLAM project ORB-SLAM2. The visual odometry we used is ORB (Oriented FAST and Rotated BRIEF), which is a combination of Key-point Oriented FAST and binary descriptor BRIEF. FAST is a quick algorithm to get the Key-points of the image. The basic step is: 1. Calculate the lightness I_p of a given pixel; 2. Set a threshold T for the lightness; 3. Let the given pixel be the centre of a circle with radius 3, and get the 16 pixels on this circle; 4. If there are N continuous pixels on the circle with lightness greater than $I_p + T$ or smaller than $I_p - T$, then the given pixel can be treated as a Key-point. N is 12 in our case. We need to implement the Non-Maximal-Suppression to keep the most significant Key-points. ORB adds the description of scale invariance and rotation. For the scale invariance, the Image Pyramid is used to make sure that the Key-points are on every level of the image pyramid. The orientation information of the Key-points is calculated based on Intensity Centroid. The moment of an image square B is defined as

$$m_{pq} = \sum_{x,y \in B} x^p y^q I(x,y), \quad p,q = \{0,1\}$$

The centroid of this image square can be calculated as $C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right)$, and we can get vector by connecting the geometric centre and the centroid. The orientation of the Key-point is described by an angle, which is $\theta = \arctan(m_{01}/m_{10})$.

To get the descriptor for the current Key-point, we use BRIEF, which use 0 and 1 to represent a pixel by comparing two random pixels near the given Key-point. If p is larger than q, then will be 1, if d is larger than. To match up the Key-points in different frames, we can simply find the Key-points for each frame and calculate their Hamming distance. After matching up the Key-points, we could calculate the pixel difference of a particular Key-point between different frames, combined with small hole model, we could estimate the displacement and rotation of the camera, and update our real time coordinates for rover.

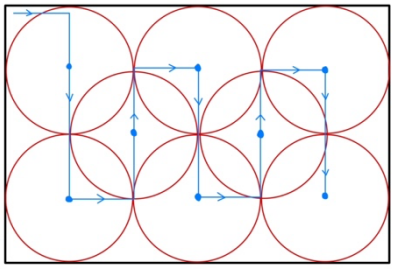
For the Backend-Optimization and Loop Back Detection, ORB-SLAM2 has a good implementation. The basic logic of the Loop Back Detection is based on BoW(Bags-of-Words) model, which is a classification of different eigen values in the Dictionary. There would be some parameters to describe an image, and we need to train our dictionary based on DBoW library and to check the similarities of these parameters for each image. In this way, we could know whether the camera comes back to its original position. ORB-SLAM2 has a detection every 10 frames and to check the significant frame. A special property of ORB-SLAM2 is that if we are disturbed in the middle and come back later, the data can be reconnected. This property is useful since sometimes we need to have a sudden 90-degree rotation to avoid the aliens, and if we returned, the entire process can be continued.
We use the SLAM to generate our 3D map, but the camera given to us is a normal monocular camera and we cannot get any spatial information and cannot generate dense cloud point image, so we add an Intel RealSense SR300 RGBD camera. We also add a cloud point generator and cloud point viewer to the original ORB-SLAM2[16]. The generated map is connected to the Web by ROS[17]. Our SLAM is running on a Raspberry Pi 4B remotely.

3.6 Control

3.6.1 Route planning

3.6.1.1 Route planning without Radar Detection

According to testing, the vision module has a maximum detection range of 90 cm and an optimal detection range between 20 and 80 cm. In order to travel the shortest distance and detect all the aliens in the area, the Rover needs to rotate 360 degrees at each blue point to search for aliens and detect colour in red circle region.



3.6.1.2 Route planning with Radar Detection

By using a similar path planning approach as before, combined with the test distance detected by radar, a path that can be traversed throughout the map is achieved. Because this path travels throughout the map, and the visual detection range is much larger than that of radar, rotating 360 degrees in the middle is redundant.

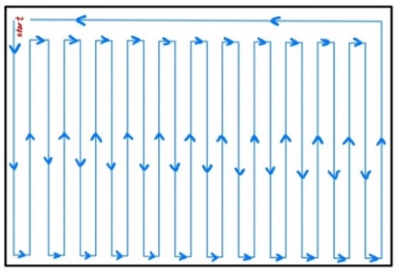


Fig 28 & 29: Route planning without & with Radar Detection

3.6.2 Avoidance

The rover moves forward 50 cm each time or rotate 90 degree if the rover reaches the turning point. During the stop, the relative x and y coordinates of alien in front of the rover can be calculated from the data retrieved from vision module and the angle the rover rotate to face the alien. If an alien or building is on the default route (absolute x coordinate is smaller than 15 cm and the absolute y coordinate is smaller 50 cm), the controller executes alien avoidance operation. The rover will turn right if the alien is on the lefthand side and vice versa. After the operation, the rover further check the route and determine whether further

3.6.3 Connection between ESP32 and Database

JSON format: data are transmitted between database and ESP-32 in JSON format as it is lightweight which requires less bytes in transmission it is relatively easy for machine to parse and generate which in return occupies less resources and guarantees faster data encoding and transmission. Also, it is human-friendly and readable which ease the process of debugging. [15]

3.6.4 Control Mode Switch

Manual Logic: The Command Module allows user to switch to remote manual control with a switch. When manual switch is on, the web application will send an integer "1" to database under node "*mode*" and this indicates that the rover should be switch to manual control ("0" for auto mode). The ESP-32 will then fetch this number from database and send command instructions to the drive module. The command interface allows user to control the rover with both on-screen arrow buttons and keyboard. Taking on-screen arrow buttons as an example, when a button is clicked, it would send an integer number to the database under node "manual_instruction/move" (see appendix)and the ESP32 retrieves data and send corresponding instructions to the drive module.

4. Testing

4.1 Energy module testing

4.1.1 Relay

After blocks were put together, the USB was plugged into the battery for charging under sunlight. The battery was charging as expected, but suddenly the light intensity surged as the cloud moved. Sudden surge in intensity caused an increase in the voltage of the PV panel, which in turn caused the output voltage of the second SMPS to surge beyond the safe range and damaged the battery. To prevent this from happening again, a relay should be adopted after the second SMPS. When the voltage is within 4.5V and 5.2V, relay is closed. If voltage is out of that range, relay acts as a switch and switch the system to a load.

4.1.2 Voltage limit of the Arduino

Port B of the SMPS is limited by 4.1V, as the Arduino board can operate no higher than that, otherwise severe damage may be caused. A potential divider is used to solve this problem, a newly assigned pin A1 is wired in the middle of two same resistors and the voltage across these two resistors is no longer limited.

4.1.3 Efficiency

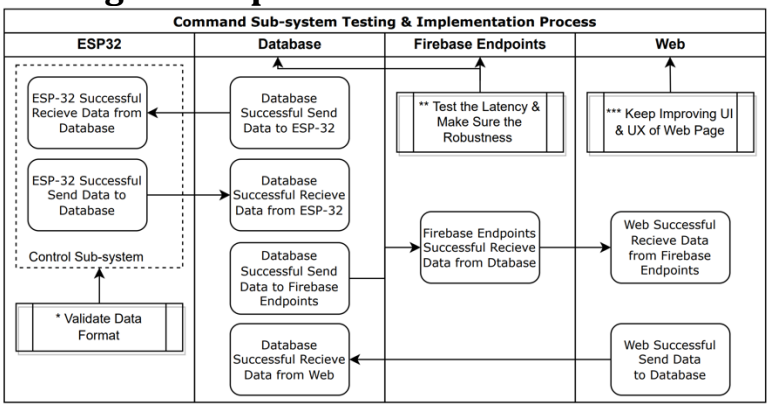
The efficiency of the system can be found by $\frac{Power\ goes\ into\ the\ battery}{Power\ yield\ by\ PV\ panels}$, the table below illustrates the efficiencies, 20 minutes between each sample.

Time	Power from PV panel	Power into the battery	efficiency
14:00	2.578W	1.575W	61.1%
14:20	1.978W	1.112W	56.2%
14:40	2.389W	1.592W	66.6%
15:00	2.420W	1.662W	68.7%
15:20	1.556W	0.780W	50.1%

4.1.4 Charging speed

The charging speed can be represented by the charging current, the higher the charging current the faster the charging process. Having a battery capacity of 5000mAh and charging current of 0.318A as shown above, the ideal charging time would be $\frac{5000mAh}{318mA} \approx 15.72h$, but there are power losses in the charging process, 40% of loss normally is considered [18]. Then the real charging time would be $\frac{5000+5000\times0.4}{318} \approx 22.01h$.

4.2 Command - Testing and implementation Procedure Flowchart



5. Conclusions

5.1 Achievement

The main objective of the project is successfully fulfilled, and every submodule can operate as expected. An extra 3D map and a two-mode manual control system are applied to extend the functionality of the rover.

5.2 Future improvements

In command sub-system's backend, firebase endpoints can be removed and accessing database directly from web app using Rest API with compromise in scaling and off-line accessibility but simplify the project structure and resources used significantly.

For energy sub-system, the first improvement can be made on the efficiency of the system, 60% efficiency is not ideal for power transmission. Thus a better SMPS that has lower MOSFET switching loss and component heat losses should be applied. The Second improvement can be made based on the SOC (state of charge) of the battery, as the battery is not charged up the way we want it to. The power bank has its own charging code that uses CC (constant current) mode first and then CV(constant voltage) mode to charge the battery. Therefore, a more sophisticated code in the second SMPS can be developed to provide better efficiency.

For the radar circuit, it would be better to increase the gain. A more precise signal can be output, and the detection range can be increased as well.

6. Project Management

To track the progress of the project, the team used a Gantt chart to make records and consistently updated it through the whole timeline. With the team meeting three times a week to update on the Gantt chart and to summarize the project.

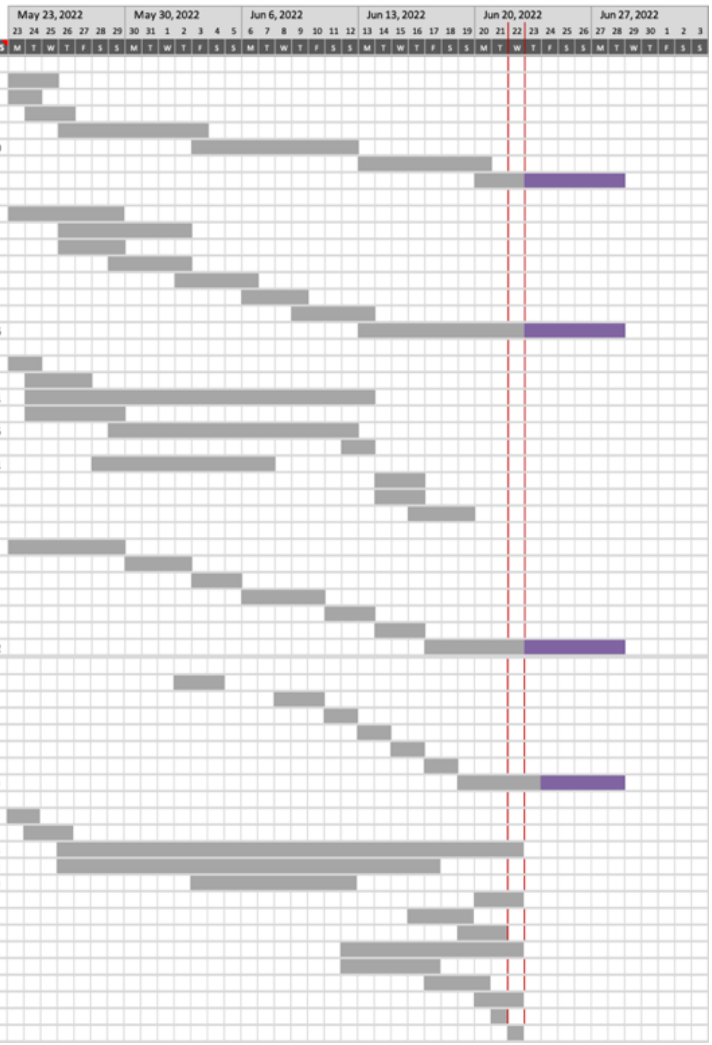
Mars Rover 2022

Group 7

Jubo Xu/ Justin Huang/ Charlie Chen/ Jacky Jiang/ David Zheng/ Ziduan Li/ Rui Li

Project Start: 2022/5/23
Today: 2022/6/22
Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END	DAYS
Energy					
1. Characterise the P-V and I-V curve of PV panels	Rui Li	100%	23/05/2022	25/05/2022	3
2. Characterise the I-V curve of the power bank	Rui Li	100%	23/05/2022	24/05/2022	2
3. Come up with a general idea of the system	Rui Li	100%	24/05/2022	26/05/2022	3
4. Design of MPPT algoritm and testing	Rui Li	100%	26/05/2022	03/06/2022	9
5. Design of battery charing profile and testing	Rui Li	100%	03/06/2022	12/06/2022	10
6. Connect MPPT and battery charging profile	Rui Li	100%	13/06/2022	20/06/2022	8
7. Data collection and further improvement	Rui Li	40%	20/06/2022	28/06/2022	9
Drive					
1. Assemble the rover &Optical Sensor Calibration	Justin Huang	100%	23/05/2022	29/05/2022	7
2. Implement Straight Line Control & Testing	Justin Huang	100%	26/05/2022	02/06/2022	8
2.1 PID Control Code Writing	Justin Huang	100%	26/05/2022	29/05/2022	4
2.2 PID Control Tunning and Testing with Position Control	Justin Huang	100%	29/05/2022	02/06/2022	5
3. Implement Position Control & Testing & Calibration	Justin Huang	100%	02/06/2022	06/06/2022	5
4. Implement Angle Control & Testing & Calibration	Justin Huang	100%	06/06/2022	09/06/2022	4
5. Command & Manual Control Testing	Justin Huang	100%	09/06/2022	13/06/2022	5
6. Integration & Control Testing	Justin Huang	65%	13/06/2022	28/06/2022	16
Command					
1. (Front & Back-end) Function Definition & Design Outline	Jacky & David	100%	23/05/2022	24/05/2022	2
2. (Front & Back-end) Log-in / Authentication Page by Firebase	David	100%	24/05/2022	27/05/2022	4
3. (Front-end) Main dashboard webpage design (ReactJS & MUI)	Jacky	100%	24/05/2022	13/06/2022	21
3.1 Coordinates dashboard with Real-time Coordinates Data	David	100%	24/05/2022	29/05/2022	6
3.2 Map with Real-time Rover & Alien & Infrastructure Icon	Jacky	100%	29/05/2022	12/06/2022	15
3.3 Resources Navigation & Real-time Rover Status	Jacky & David	100%	12/06/2022	13/06/2022	2
4. (Back-end) Connect main dashboard and local database (Post & Get - NodeJS)	David	100%	28/05/2022	07/06/2022	11
6. (Front & Back-end) Migrate to Firebase database	Jacky	100%	14/06/2022	16/06/2022	3
7. (Front-end) Mannual control panel	Jacky	100%	14/06/2022	16/06/2022	3
8. Integrate & Testing with Drive	Jacky & David	100%	16/06/2022	19/06/2022	4
Radar					
1. Learn about doppler effect and the check the property of HB100	Ziduan Li	100%	23/05/2022	29/05/2022	7
2. Collect and analyse the data of original IF signal	Ziduan Li	100%	30/05/2022	02/06/2022	4
Design and test the amplifier circuit	Ziduan Li	100%	03/06/2022	05/06/2022	3
Design and test the band-pass filter	Ziduan Li	100%	06/06/2022	10/06/2022	5
Design and test the peak detector	Ziduan Li	100%	11/06/2022	13/06/2022	3
Connect the circuit with ESP32 and locate the target	Ziduan Li	100%	14/06/2022	16/06/2022	3
Improve the accuracy of target location	Ziduan Li	50%	17/06/2022	28/06/2022	12
Control					
1. Connect ESP-32 with WIFI	Jacky Jiang	100%	02/06/2022	04/06/2022	2
2. Connect ESP-32 with Firebase Realtime Database	Jacky Jiang	100%	08/06/2022	10/06/2022	3
3. Rover manual control testing	Charlie Chen	100%	11/06/2022	12/06/2022	2
4. Establish connection between ESP-32 with FPGA	Bob Xu	100%	13/06/2022	14/06/2022	2
5. Establish connection between ESP-32 with radar	Ziduan Li	100%	15/06/2022	16/06/2022	3
6. Route Planning	Charlie Chen	100%	17/06/2022	18/06/2022	3
7. Avoidance	Justin Huang	50%	19/06/2022	28/06/2022	10
Vision					
1. Build the starter project	Charlie Chen	100%	23/05/2022	24/05/2022	2
2. Reasearch and finish the RGB to HSV Convertor	Charlie Chen	100%	24/05/2022	26/05/2022	3
3. Color Detection and Pixel Processing	Charlie Chen	100%	26/05/2022	22/06/2022	28
3.1 Finding the HSV Region for six Colours	Charlie Chen	100%	26/05/2022	17/06/2022	23
3.2 3x3 Kernel Filter	Jubo Xu	100%	03/06/2022	12/06/2022	10
3.3 Cutting the Upper Sreen	Charlie Chen	100%	20/06/2022	22/06/2022	3
4. Draw the bounding boxes	Charlie Chen	100%	16/06/2022	19/06/2022	4
5. Distance and Pixel Width Measurement	Charlie Chen	100%	19/06/2022	21/06/2022	3
6. Building Detection	Jubo Xu	100%	12/06/2022	22/06/2022	11
6.1 Contrast improvement	Jubo Xu	100%	12/06/2022	17/06/2022	6
6.2 Black and White pixel extraction	Jubo Xu	100%	17/06/2022	20/06/2022	4
6.3 Edge detection and bounding box generation	Jubo Xu	100%	20/06/2022	22/06/2022	3
7. UART Bits Design and Processing	Charlie Chen	100%	21/06/2022	21/06/2022	1
8. Integrate & Testing with Control	Charlie Chen	100%	22/06/2022	22/06/2022	1



7. References

[1] A. Dolara, G. Lazaroiu, S. Leva and G. Manzolini, "Experimental investigation of partial shading scenarios on PV (photovoltaic) modules", *Energy*, vol. 55, pp. 466-475, 2013. Available: 10.1016/j.energy.2013.04.009.

[2] *Youtube.com*, 2022. [Online]. Available: https://www.youtube.com/watch?v=0ItjKs7aJFM&t=1012s. [Accessed: 21- Jun- 2022].

[3] Blog.csdn.net. 2022. FPGA 实现 RGB 与 HSV 的转换_满城風絮的博客-CSDN 博客_fpga rgb 转. [online] Available at: <https://blog.csdn.net/qq_41527741/article/details/121412676?ops_request_misc=&request_id=&biz_id=102&utm_term=verilog%E5%AE%9E%E7%8E%B0%20rgb%20%E8%BD%AC%20hsv&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-1-121412676.142^v20^pc_rank_v35,157^v15^new_3&spm=1018.2226.3001.4187> [Accessed 22 June 2022].

[4] Docs.couchbase.com. 2022. Admin REST API | Couchbase Docs. [online] Available at: <https://docs.couchbase.com/sync-gateway/current/rest-api-admin.html> [Accessed 22 June 2022].

[5] Expressjs.com. 2022. Express - Node.js web application framework. [online] Available at: <https://expressjs.com/> [Accessed 22 June 2022].

[6] Firebase. 2022. Admin SDK Reference | Firebase. [online] Available at: <https://firebase.google.com/docs/reference/admin> [Accessed 22 June 2022].

[7] Firebase. 2022. Authorization | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/auth> [Accessed 22 June 2022].

[8] Firebase. 2022. Cloud Firestore | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/firestore?hl=en&authuser=0> [Accessed 22 June 2022].

[9] Firebase. 2022. Cloud Functions for Firebase | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/functions> [Accessed 22 June 2022].

[10] Firebase. 2022. Firebase Realtime Database | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/database?hl=en&authuser=0> [Accessed 22 June 2022].

[11] Firebase. 2022. Introduction to the Admin Database API | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/database/admin/start?hl=en&authuser=0> [Accessed 22 June 2022].

[12] Firebase. 2022. Structure Your Database | Firebase Documentation. [online] Available at: <https://firebase.google.com/docs/database/admin/structure-data?hl=en&authuser=0> [Accessed 22 June 2022].

[13] Fullstackfirebase.com. 2022. What is serverless? - Full-Stack Firebase. [online] Available at: <https://www.fullstackfirebase.com/introduction/what-is-serverless> [Accessed 22 June 2022].

[14] "Making the electronics for a $7 USD doppler motion sensor - Limpkin's blog", *Limpkin.fr*, 2022. [Online]. Available: https://www.limpkin.fr/index.php?post/2013/08/09/Making-the-electronics-for-a-%247-USD-doppler-motion-sensor. [Accessed: 22- Jun- 2022].
[15] "JSON", Json.org, 2022. [Online]. Available: https://www.json.org/json-en.html. [Accessed: 22- Jun- 2022].
[16]"GitHub - gaoxiang12/ORBSLAM2_with_pointcloud_map", *GitHub*, 2022. [Online]. Available: https://github.com/gaoxiang12/ORBSLAM2_with_pointcloud_map. [Accessed: 22- Jun- 2022].
[17]"GitHub - appliedAI-Initiative/orb_slam_2_ros: A ROS implementation of ORB_SLAM2", *GitHub*, 2022. [Online]. Available: https://github.com/appliedAI-Initiative/orb_slam_2_ros. [Accessed: 22- Jun- 2022].
[18] *ADATA Technology – Innovating the Future*, 2022. [Online]. Available: https://corp.adata.com/us/support/quiktips/conversion-efficiency. [Accessed: 22- Jun- 2022].

8. Appendices

8.1 PV panels

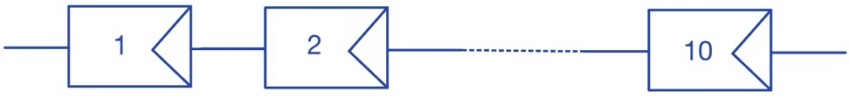


Fig 1: 10 cells in series for one PV panel

8.2 MPPT

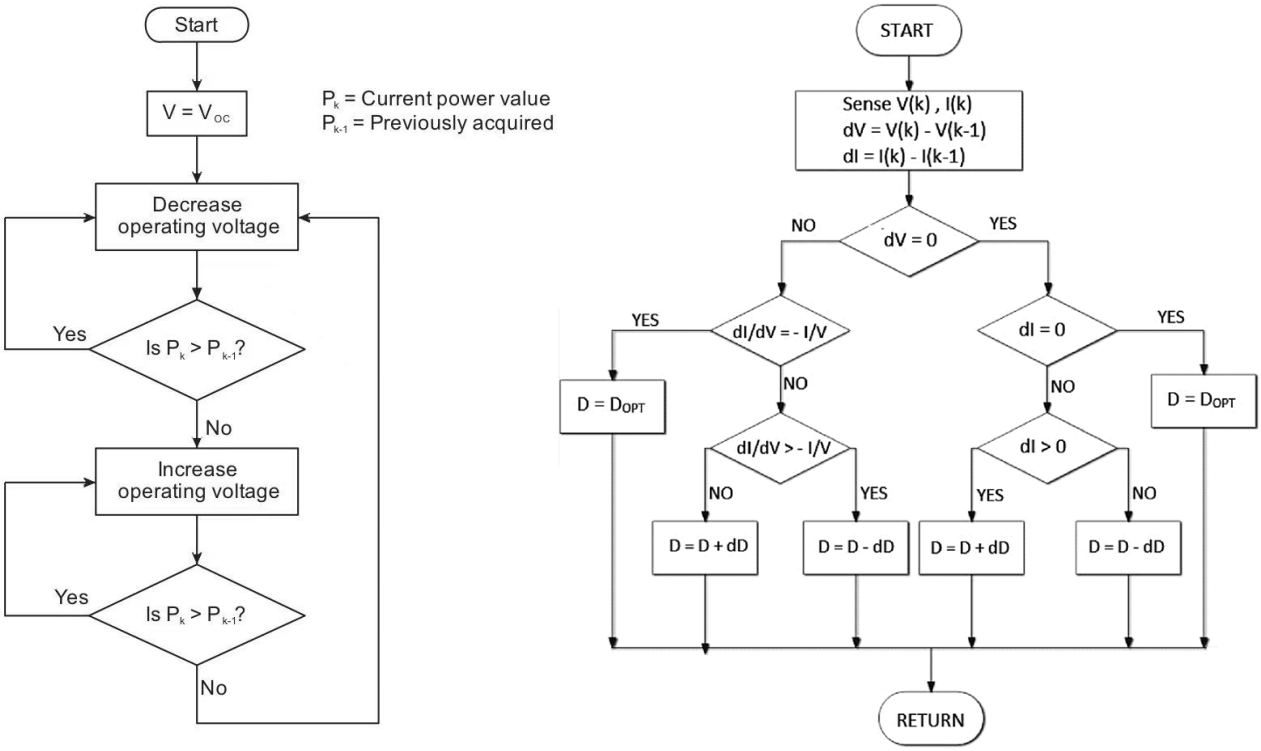


Fig 2: Perturb and observe algorithm(left) and Incremental conductance algorithm(right)

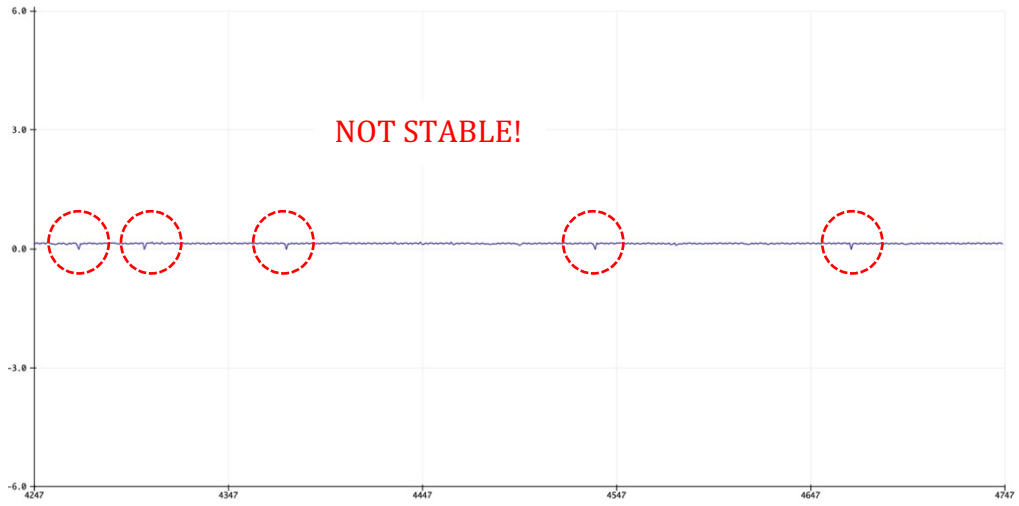


Fig3: MPP detection using perturb and observe algorithm

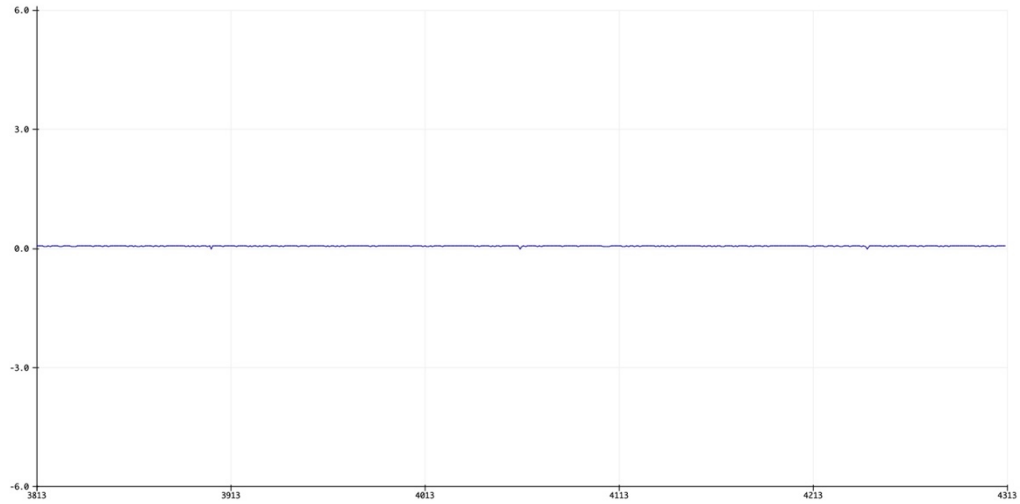
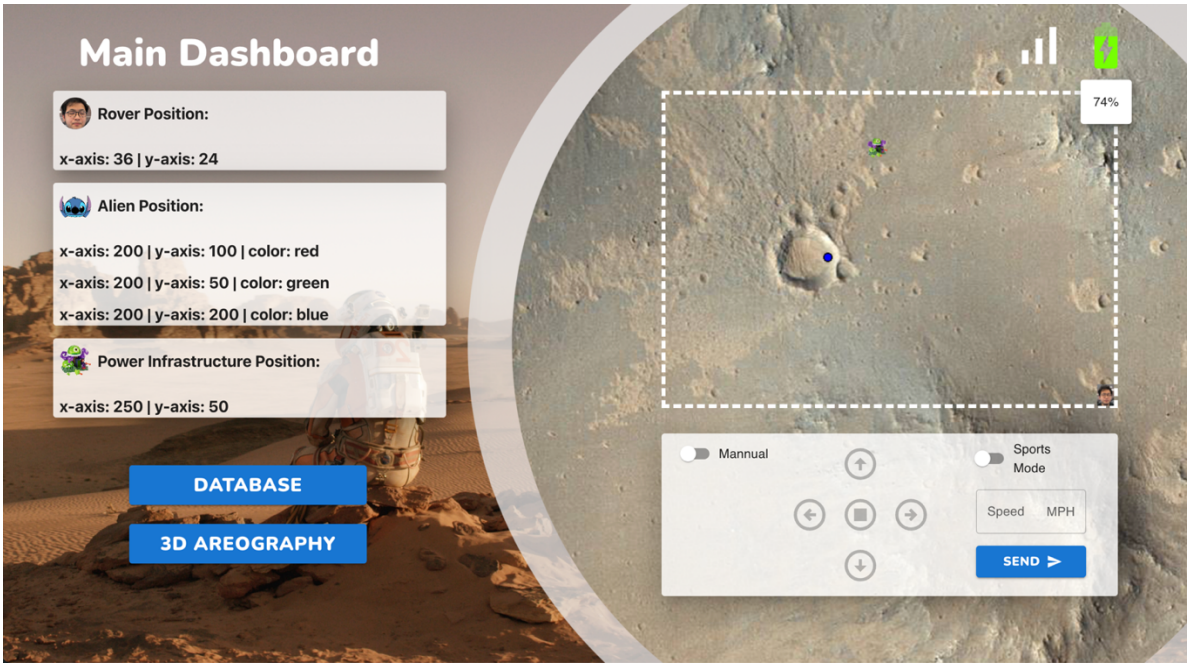


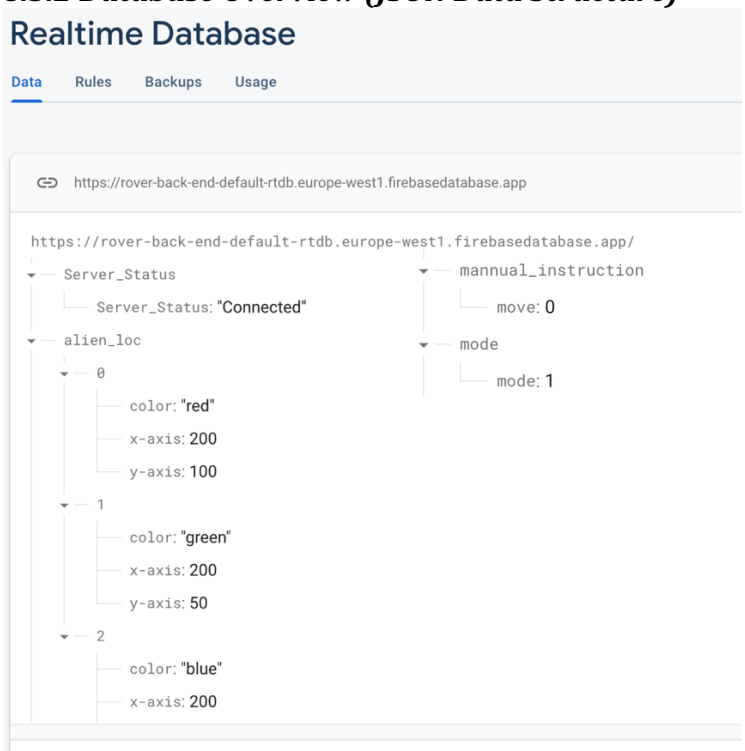
Fig4: MPP detection using incremental conductance algorithm

8.3 Command

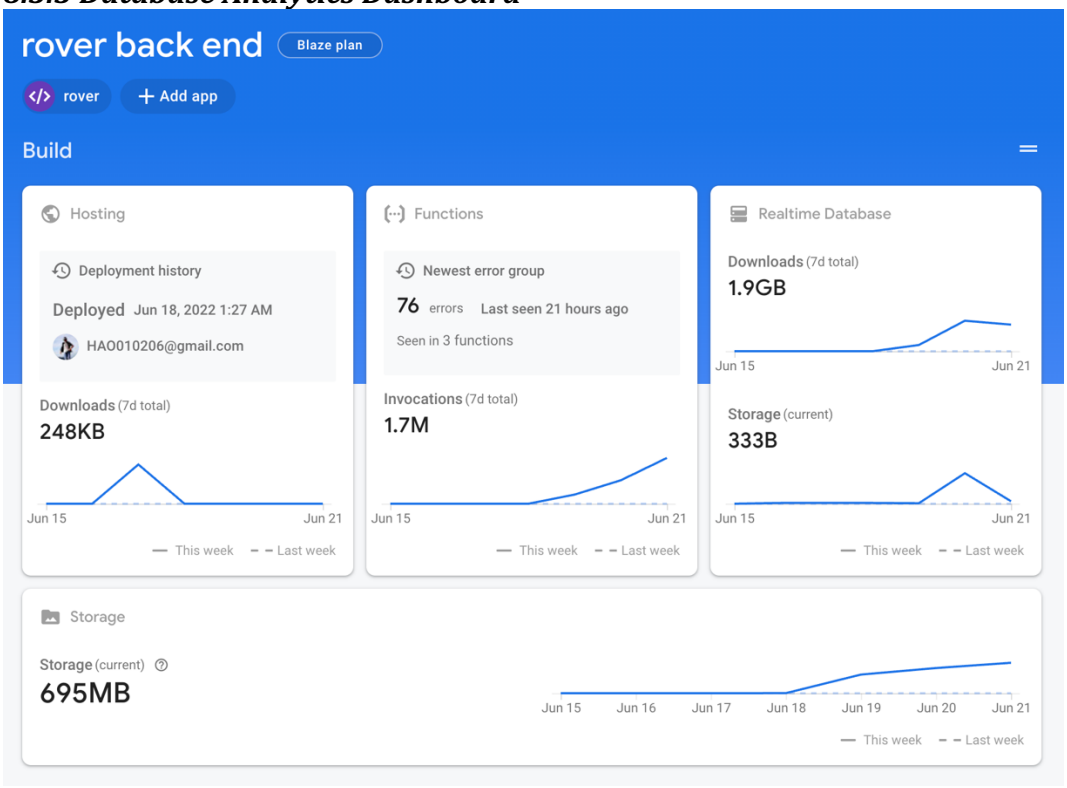


8.3.1 Webpage Overview

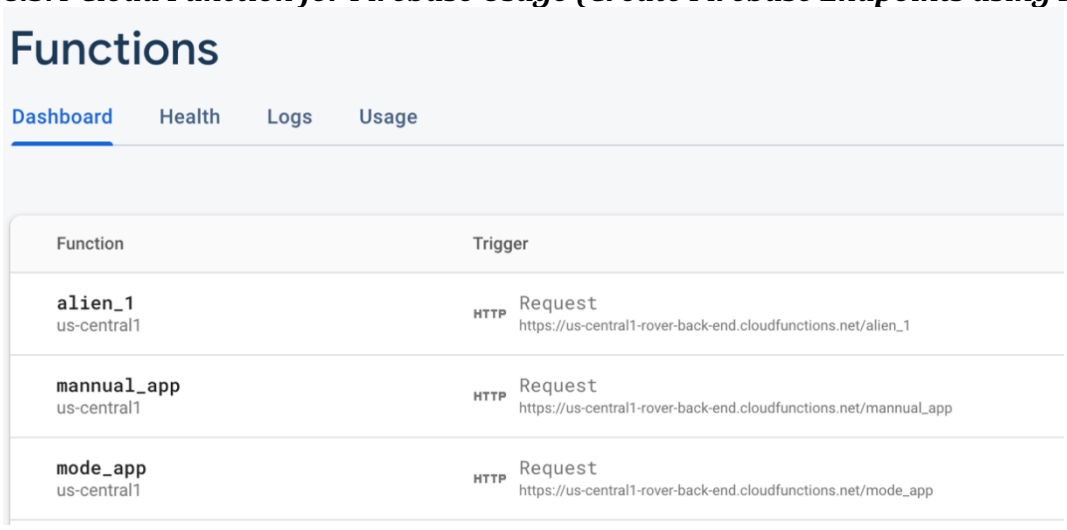
8.3.2 Database Overview (JSON Data Structure)



8.3.3 Database Analytics Dashboard



8.3.4 Cloud Function for Firebase Usage (Create Firebase Endpoints using HTTP)



8.4 Radar

8.4.1 Threshold voltage value

```
src > main.cpp > loop()
1   #include <Arduino.h>
2
3   void setup() {
4     // put your setup code here, to run once:
5     Serial.begin(115200);
6   }
7
8
9
10  void loop() {
11    // put your main code here, to run repeatedly:
12    Serial.println(analogRead(25));
13    delay(250);
14
15  }
```

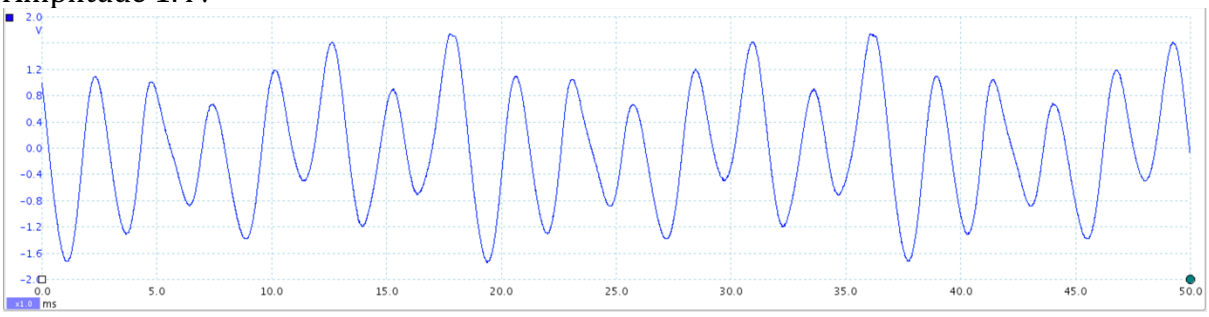
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
954
937
941
934
944
945
938
941
950
933
933
```

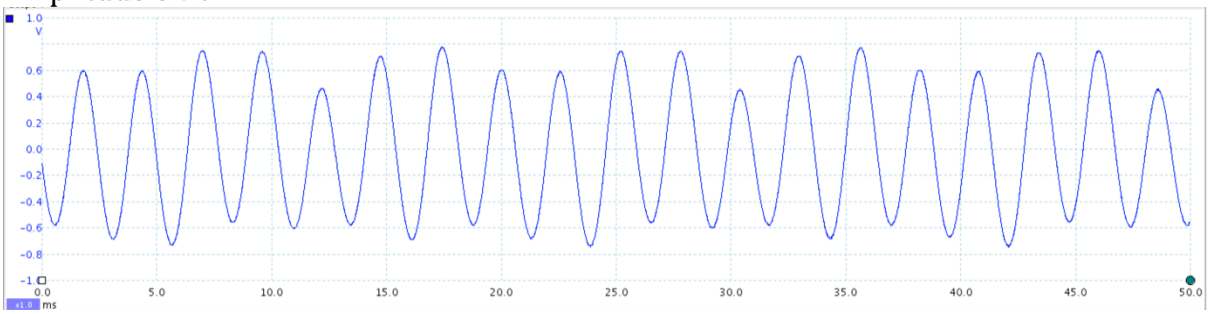
(Above 900 when radar is right above target)

8.4.2 Signal strength at different angle

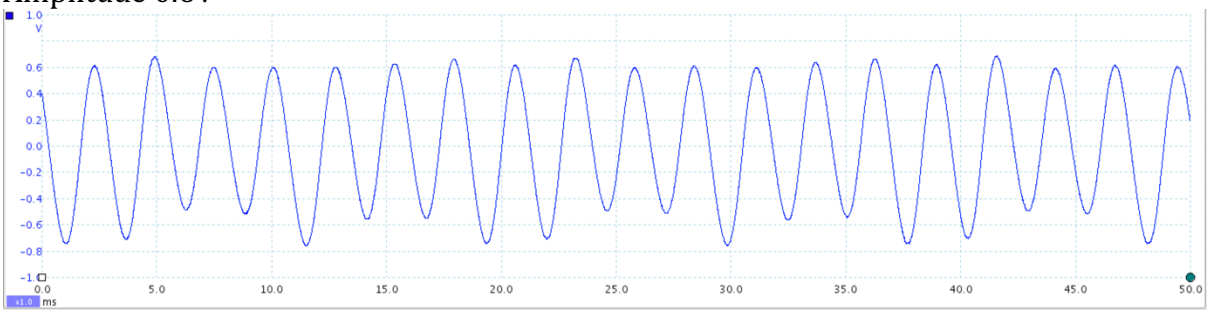
0-degree
Amplitude 1.4V



22-degree
Amplitude 0.7V



27-degree
Amplitude 0.6V



40-degree
Amplitude 0.2V