Imperial College London

Department of Electrical and Electronic Engineering

Engineering Design Project: Mars Rover

Authors	CID
Cathy Ding	01856648
Feng Shen Foo	01882052
Weizheng Wang	01865672
Tszhang Wong	01864937
Haoran Wu	01954226
Mengyuan Yin	01842827

Submitted to Dr. Adam Bouchaala

May-June 2022

ABSTRACT

In this project, a fully-integrated autonomous Mars rover system is built. The rover can navigate through an arena, detecting aliens, buildings and underground infrastructure along the path exhaustively. The rover is powered by batteries charged at a power station with photovoltaic cells. Maximum power point tracking and state of charge estimation algorithms are implemented for energy monitoring. A hybrid on-off and proportional controller, with feedback from optical flow sensing, is implemented to drive the rover and track its position and angle. To ensure the accuracy of position tracking, ultrasound and infrared devices are added for error detection and correction purposes. During rover exploration, a Doppler radar followed by an analogue amplification and filtering circuit is used to detect underground power stations. A camera is used for the identification of aliens and buildings. Pre-processing filters are firstly applied on the frames captured for noise removal. Since buildings have white and black stripes, a 1D refined edge detection method based on HSV is applied to decompose the building into individual stripes. Then, self-designed algorithms are applied to extract aliens and buildings information from HSV data. Situations where multiple objects are presented within one image are also well-considered. For endpoint users to monitor rover status and collected information, a web app is developed. A database server and a web server are set up on the AWS cloud to allow data exchange.

Contents

1 Overall Design and Architecture

In order to divide and conquer the project, an initial requirement diagram is drawn to identify functionalities required by the Mars rover system, as shown in Fig. 1. For non-functional requirements, the rover needs to complete the exploration of the arena within a reasonable amount of time. The system also needs to be easy to use and have a relatively short end-to-end communication latency to keep the rover responsive.

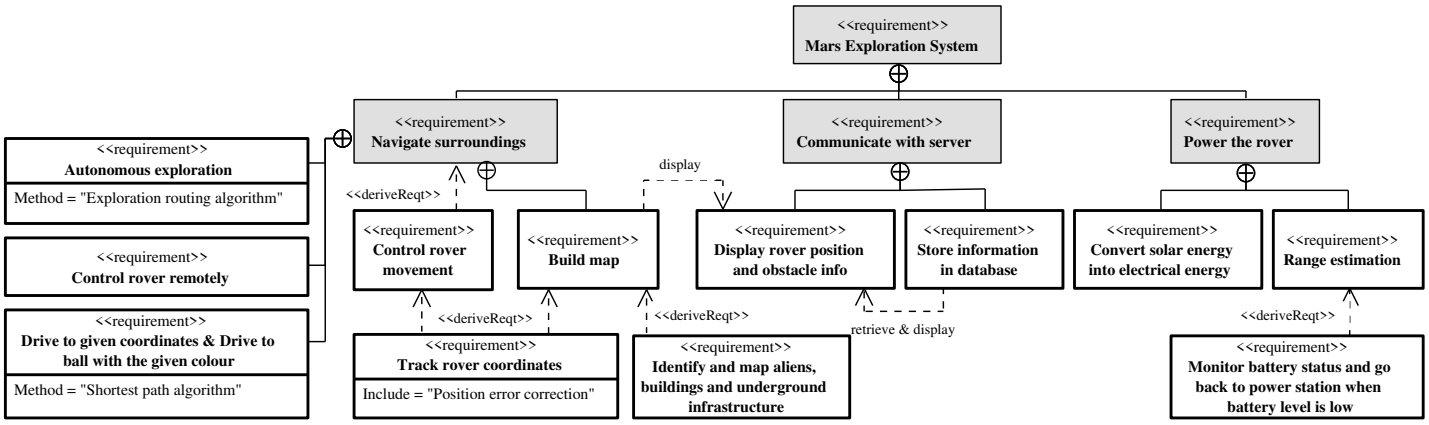


Fig. 1: *Requirement diagram of the project*

During the designing stage, a high-level structural diagram is drawn showing the overall architecture and the communication structure of the Mars rover system, as shown in Fig. 2.

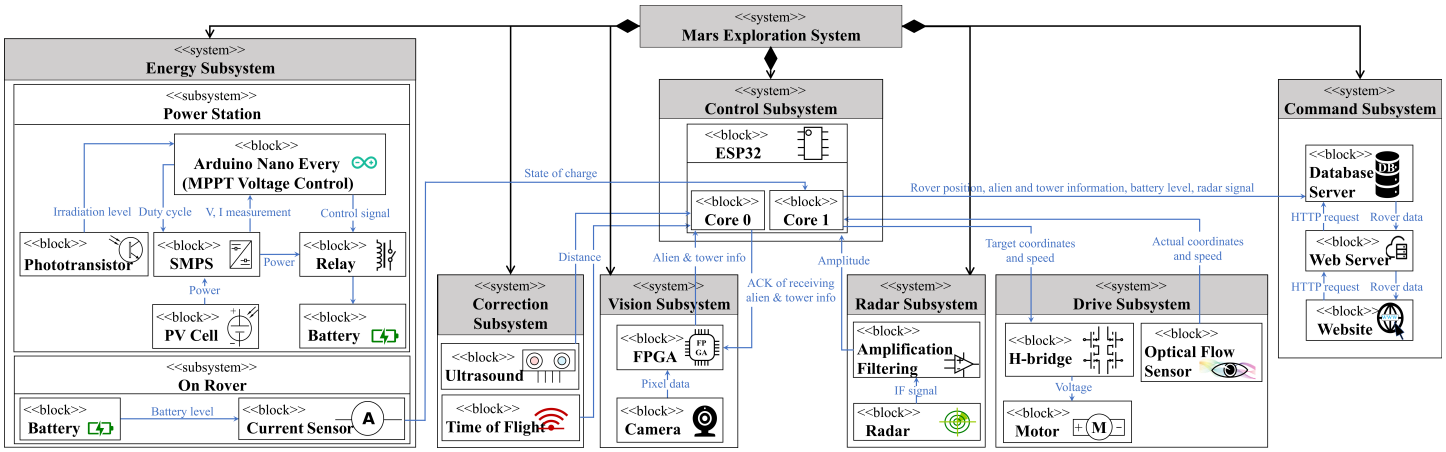


Fig. 2: *Block definition diagram showing the overall structure*

2 Project Management

To facilitate group collaboration, a progress management chart was created to keep track of the project schedule. Daily group meetings were conducted to update working progress and discuss plans.

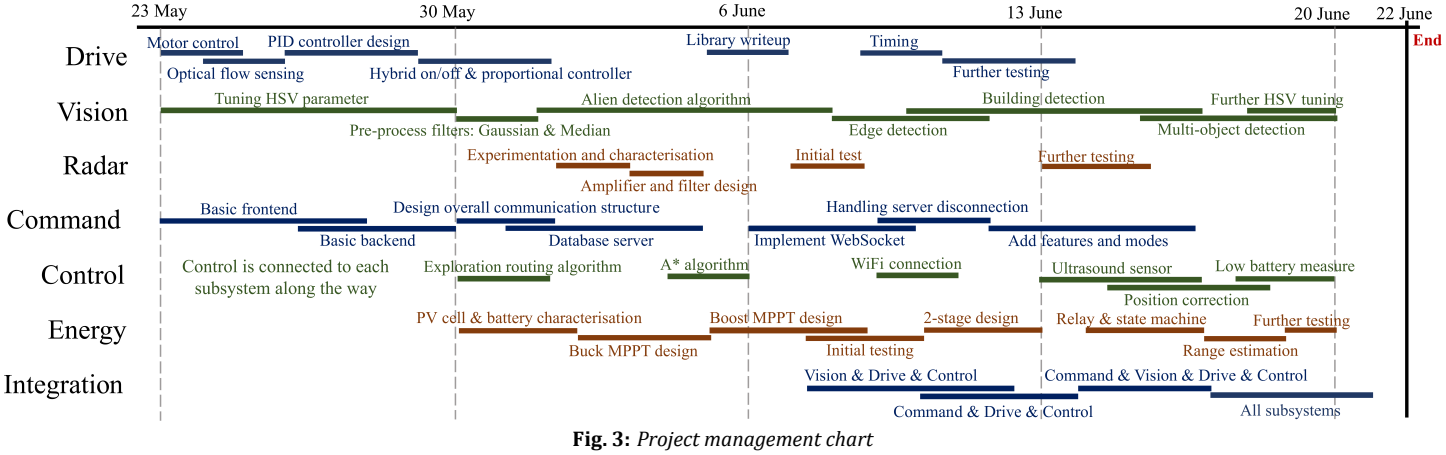


Fig. 3: *Project management chart*

Energy

The energy subsystem aims to convert solar energy to electrical energy to charge the batteries for the rover.

3.1 Theory

The current-voltage (IV) characteristics for a photovoltaic (PV) cell and a battery are respectively given by:

$$I = I_{\text{SC}} - I_{\text{S}}(e^{\frac{V}{V_T}} - 1) \qquad V = E - IR_{\text{ESR}}$$

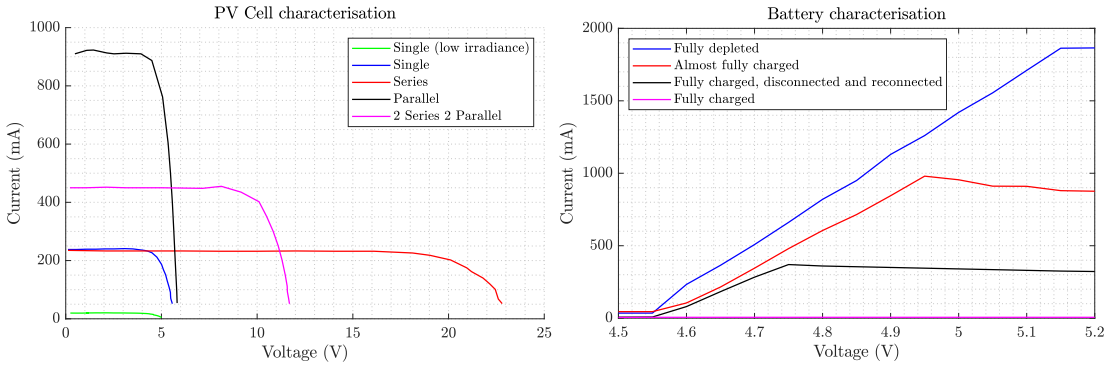
where I_{SC}: current under short circuit conditions, V_T: thermal voltage, and I_{S}: reverse saturation current

where E: electromotive force (EMF), R_{ESR}: internal resistance

The maximum power point (MPP) is the global maximum power that the PV cell can provide. [1] Maximum power point tracking (MPPT) is the process of tracking the MPP on the power-voltage curve. MPPT can be achieved using switch mode power supplies (SMPS) to match the battery load to the MPP.

Impedance referral is used to analyse the IV characteristics [2] seen from the input and output. For an input voltage V_1 and an output voltage V_2, and for $k_V = \frac{V_2}{V_1}$, assuming power conservation, the referred voltage is $V_2' = \frac{V_2}{k_V}$ and the referred impedance is $Z_2' = \frac{Z_2}{k_V}$.

3.2 PV cells and battery characterisation



(a) *IV characteristic of different arrangements of PV cells* **(b)** *IV characteristic of battery*

Fig. 4: *IV characteristics measurements*

The IV characteristics of PV cell(s) in different arrangements (single, 4 series (4S), 4 parallel (4P), 2 series 2 parallel (2S2P)) as well as the battery are shown in Fig. 4b. The main observations are:

- In low light level (indoor), the current is significantly smaller than in strong sunlight (outdoors).
- Cells arranged in series have larger voltages whereas cells arranged in parallel have larger currents.
- The built-in battery management system (BMS) limits the current to $1.8\,\text{A}$, or much lower (down to $10\,\text{mA}$) when the battery is close to fully charged.

Partial shading has little effect on PV cells in parallel but has a significant effect on PV cells in series, since the shaded panel impedes the current flow through the overall circuit when connected in series. Bypass capacitors are tested to be effective in reducing the effect of partial shading (Appendix fig. 5).

3.3 Design Process

The design process involves understanding the IV characteristics of the PV cells and the battery, limitations in hardware, and testing the SMPS in open-loop.

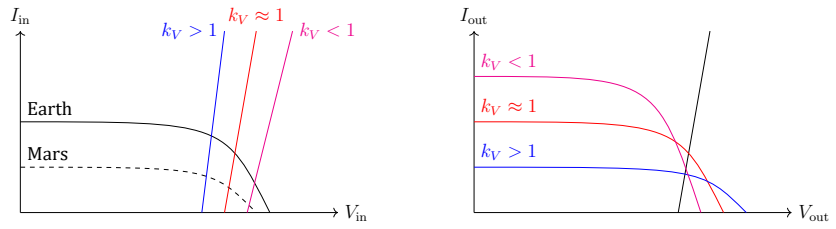


Fig. 5: *Impedance referral graphs for battery and PV cells*

The IV graph from the input side is sketched by applying impedance referral on the battery (Fig. 5). Assuming good lighting conditions, the MPP is estimated to be not far left from the battery load line. If the solar irradiance on Mars ($590\,\text{W}\,\text{m}^{-2}$ in comparison to around $1000\,\text{W}\,\text{m}^{-2}$ on Earth) is considered, the MPP will

be shifted towards the left even further. Hence, in theory, the condition $k_V = \frac{V_{\text{out}}}{V_{\text{in}}} > 1$ must be met.

Hardware limitations include voltage limits in synchronous Buck and Boost SMPS, the quantity and position of current sensors, and the quantity of Arduino adapters provided.

Consequently, some potential system layouts can be concluded non-feasible:

PV cell	SMPS	Feasible (?)	Comments
4S, 2S2P	-	✗	SMPS voltage limit
4P	Buck S/NS	✗	Could only work in high irradiance levels (Refer fig. 5)
4P	Boost S	✓	Pros: Highest efficiency Cons: Cannot do MPPT
4P	Boost NS	✓	Pros: High efficiency with MPPT Cons: Could not access MPP region in high irradiance levels
4P	Buck-Boost S	✓	Pros: High efficiency with more robust MPPT Cons: No access to input and output power
4P	Boost-Buck S	✓ (Final design)	High efficiency with more robust MPPT Can measure input and output power easily

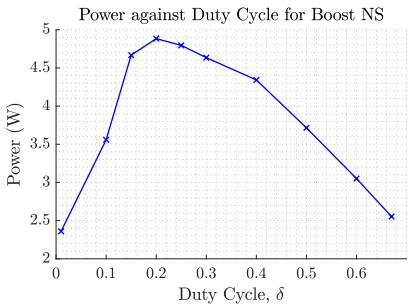
(S: synchronous; NS: non-synchronous)

3.4 Experimentation with different system layouts

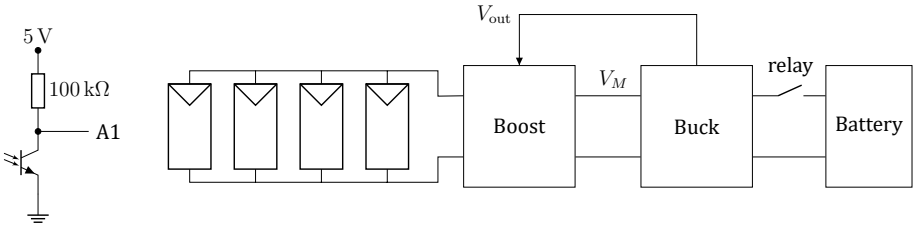
All 4 feasible designs were further tested (outdoor with high irradiance):

- **Boost S** charges the battery with duty cycle, $\delta = 0$. This is equivalent to a direct connection across the SMPS and hence the high efficiency. The MPPT could not work because it cannot access the MPP region that is on the right side of the battery IV curve. Another shortcoming is that with high irradiance level the battery could not be safely plugged in because $V_{\text{out}} > 5.2\,\text{V}$.
- With **Boost NS**, the MPP is around $\delta = 0.2$ (Fig. 6a). It is different from Boost S because of the voltage drop of around $0.7\,\text{V}$ across the diode when in NS mode. This characteristic caused the $\frac{V_{\text{out}}}{V_{\text{in}}}$ ratio to be < 0 when $\delta = 0$ hence enables the MPPT to work. However, it is discovered that when the battery is close to fully charged, extra logic is required to regulate the output voltage and limit the current so that it can continue to charge with a much lower current. Note: With the fact that the solar irradiance on Mars is around half when compared to Earth, both systems with only Boost would work well because lower irradiance level corresponds to smaller current and voltage as shown in Fig. 5 (left).
- **Buck-Boost S** charging system works well with reasonably high efficiency. However, it was not considered due to the absence of current sensor at the input and output.
- For **Boost-Buck S**, the overall system is more robust as with the Buck, the IV characteristic of the PV cells can be shifted so that the open circuit voltage of the PV cell matches the maximum output voltage allowed for charging of the battery. As a result, it is easier to maintain the output voltage when the battery is close to fully charged.

The speed of charging depends on the power ratings of the panels and the efficiency of the system. The power ratings for each panel is constant for all 4 designs. The efficiencies for different duty cycles was collected and then averaged (Boost NS: 81.6%; Boost-Buck S: 84.4%; Buck-Boost S: 83.9%; Boost S: 90.8%). Single Boost S is the most efficient, and other designs have similar efficiencies. Note that these measurements could be prone to errors due the environmental changes and reaction time.



(a) *Open loop Boost NS test*



(b) *Phototransistor circuit and top level design of charging circuit*

Fig. 6: *(a) Example open loop test and (b) Top level of final design*

3.5 Final design

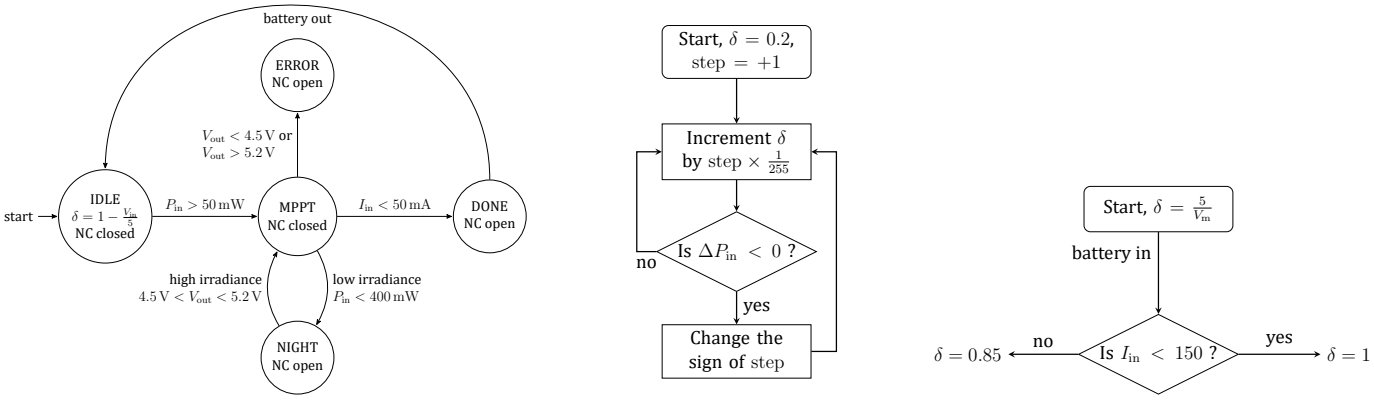
The top level of the final design is as shown in Fig. 6b. Boost-Buck S is chosen because it is highly efficient, simple to implement and has the flexibility to access both sides of the unreferred battery load line.

A phototransistor is used to detect changes in irradiance to which human might be insensitive. A dimensionless scale from 0 (maximum sunlight) to 1023 (strongly shaded) is used.

3.6 Charging algorithm

The charger is implemented with 5 states on the Boost S: IDLE, MPPT, DONE, NIGHT and ERROR (Fig. 7a).

- **IDLE**: Duty cycle is adjusted to ($5\,\mathrm{V}$) in preparation for battery to be safely plugged in.
- **MPPT**: When the battery is plugged in, the Perturb & Observe (P&O) algorithm tracks the MPP (refer fig. 7b). If the V_{out} goes beyond the range due to MPPT, the change is reverted to maintain V_{out} between $4.55\,\mathrm{V}$ and $5.15\,\mathrm{V}$ (defined as soft limits). When the battery is close to fully charged, the BMS limits the power (Fig. 8b), causing $V_{\text{out}} > 5.2\,\mathrm{V}$. In this situation, revert algorithm is replaced by another algorithm which regulates the output voltage by decreasing the duty cycle.
- **DONE**: Since the BMS cause a rapid decrease in current due to its power limiting characteristic, a current threshold is used to detect a fully charged battery. When detected, it will disconnect the circuit.
- **NIGHT**: When the input power and irradiance becomes low (detected by photodiode), the relay disconnects the circuit. It transitions back to MPPT state if the irradiance increases and input voltage regulated within range.
- **ERROR**: Immediately disconnects the circuit to protect the battery.

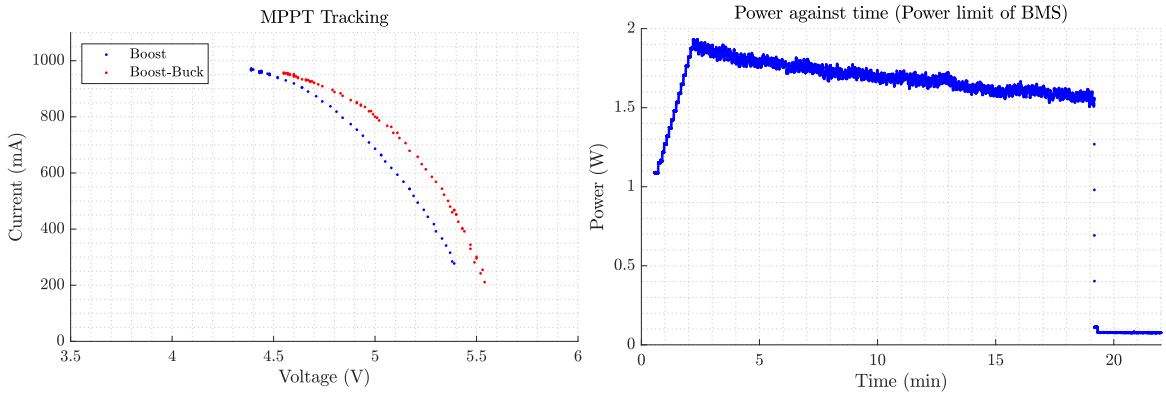


(a) *Power electronics interface state machine* **(b)** *MPPT algorithm* **(c)** *Buck logic*

Fig. 7: *Charging algorithm*

With the Buck in the second stage, the impedance seen by the Boost can be adjusted. The logic used in our design is as shown in Fig. 7c. Fast loop ($T_s = 0.01\,\mathrm{s}$) is used for state machine, calculating moving average, regulating soft limits and output voltage, whereas slow loop ($T_s = 1\,\mathrm{s}$) is used for MPPT, relay switching and corner state cases.

For the MPPT, the P&O and Incremental Conductance (IC) algorithm were both tested. Experimental results showed similar tracking speed and effects of change in irradiance level for both algorithms (Appendix fig. 26b). This could be due to difficulty of keeping the testing conditions constant. The P&O algorithm is chosen over IC because it is easier to implement and improve (Fig. 7b). The test results of MPPT and state transitions are shown in Fig. 8a and Table 1.



(a) *MPPT tracking for Boost and Boost-Buck* **(b)** *Power limit by the BMS*

Fig. 8: *MPPT tracking and power limiting graph*

Time (min)	State	Irradiance	δ	$V_{\text{in}}(\text{V})$	$I_{\text{in}}(\text{mA})$	$P_{\text{in}}(\text{V})$	$V_{\text{m}}(\text{V})$	$V_{\text{out}}(\text{V})$
0.5	idle	3	0	5.77	0	67	5.77	5
0.5	mppt	3	0	5.75	50.2	111	5.73	4.94
3.9	mppt	7	0.27	4.86	914.5	4445	6.22	4.92
4.1	mppt	1023	0.28	3.94	73.2	431	5.37	4.53
4.1	night	1023	0.28	3.89	71.6	333	5.37	4.53
4.7	night	4	0	5.85	15.9	85	5.85	5.86
4.8	mppt	4	0	5.87	11.6	67	5.86	5.01
5.0	mppt	7	0.02	5.71	262.5	1499	5.64	4.66
5.1	mppt	7	0.04	5.69	306.1	1736	5.69	4.69
19.2	mppt	4	0.05	5.58	280	1557	5.8	4.77
19.2	done	4	0.05	5.76	18.8	111	6.03	5.14

Table 1: *Test results that show state transitions*

Improvements were made to tackle some problems (Table 2) :

Problem	Solution
Voltage and inductor current ripple is non-negligible	Moving average implemented using arrays.
Current sensor giving out random zeros	Discard data when current is 0 using if statements & boolean functions.
Slow tracking when irradiance is high	Implement varying duty cycle step size

Table 2: *Improvements for charging algorithm*

3.7 Battery state of charge (SOC) and rover range estimation

An ACS711 current sensor is used to measure the current from the battery to the rover system. The sensor is Hall effect, to minimise the reduction in voltage, and a difference amplifier is used to increase the resolution of measurements of currents around $1\,\text{A}$. The SOC of the battery is estimated using Coulomb counting, $\text{SOC} = \text{SOC}_{\text{initial}} - \frac{1}{C_{\text{battery}}} \int_0^t I(t)\,dt$, and the range of the rover is then $R_{\text{rover}} = \frac{C_{\text{battery}} \times \text{SOC} \times v_{\text{rover}}}{I}$.

Components	Voltage (V)	Current (mA)	Power (W)
LEDs	4.81	0.34	1.635
Radar	4.98	0.052	0.259
FPGA	4.87	0.39	1.899
OFS	4.99	0.02	0.1
ESP32	4.97	0.07	0.348

Table 3: *Power consumption of each components*

The range can also be estimated with a time-based model, based on the measured typical power consumption of each component, as shown in Table 3.

4 Drive

The drive subsystem aims to control and track the rover's movements during navigation.

4.1 DC motor control with H-bridge

The rover's wheels are controlled by DC motors. In steady state, the angular velocity of the shaft is proportional to the voltage across the armature. A pulse-width modulation (PWM)-controlled H-bridge controls the magnitude of angular velocity by changing the duty cycle (hence the effective DC voltage), and controls the sign of angular velocity by changing the polarity of the supply. From a doublet response under no load condition with video tracking, the voltage to angular velocity transfer function was found to have first order characteristics, with static gain $\mu = 1.6\,\text{rad}\,\text{s}^{-1}\,\text{V}^{-1}$ (with $5\,\text{V}$) and time constant $\tau = 0.13\,\text{s}$.

4.2 Tracking with optical flow sensor

Local: $\begin{cases} dr = dy' \\ d\varphi = \frac{dx'}{D} \end{cases}$ where D is the distance from the OFS to the center of rotation

Global: $\begin{cases} d\theta = d\varphi \\ dx = dr\cos\theta \\ dy = dr\sin\theta \end{cases}$

The rover has two translational degrees of freedom (DOFs) and one rotational DOF, hence its state space is $\mathbb{R}^2 \times S_1$. The ADNS-3080 optical flow sensor (OFS) tracks changes by considering the flow of visible features in different frames. The OFS local (dx', dy') changes are converted to the rover's local polar $(dr, d\varphi)$

changes, which are then converted to the rover's global $(dx, dy, d\theta)$ changes. The latter are added to obtain the time evolution of the rover in state space. The rover is an underactuated system, since its current angle θ determines the next possible translations in x and y.

4.3 State space model

The rover system is defined to have inputs $\delta_L, \delta_R \in [-1, 1]$, with $|\delta_L|, |\delta_R|$ the duty cycles and $\operatorname{sgn}\delta_L, \operatorname{sgn}\delta_R$ the polarities of the H-bridge. The ideal transfer function from these inputs to the linear and angular velocities of the rover are then:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \frac{8}{0.13s + 1} \begin{bmatrix} \frac{K_r R}{2} & \frac{K_r R}{2} \\ -\frac{K_\varphi R}{2D} & \frac{K_\varphi R}{2D} \end{bmatrix} \begin{bmatrix} \delta_L \\ \delta_R \end{bmatrix} \tag{1}$$

where R is the radius of the wheel and $K_r, K_\varphi \in (0, 1)$ are gains accounting for the reduction of the shaft due to torque requirement depending on rover mass and surface friction.

A PID controller was attempted to control the local distance and angle simultaneously, using a decoupling matrix to remove cross-coupling and frequency-separated dual loops for velocities and coordinates respectively. The control structure in Fig. 9 was tuned in simulation, discretised and implemented in code. The idea was to give a (r, φ) reference which varies with time to control the rover movement. However, due to strong and difficult-to-model non-linear effects, such as static friction at low velocities and motor speed saturation limits, this design was found to be lacking robustness. As a result, it was replaced by a simpler logic, which turns out to be both more easily implementable and more suitable for exploration purposes.

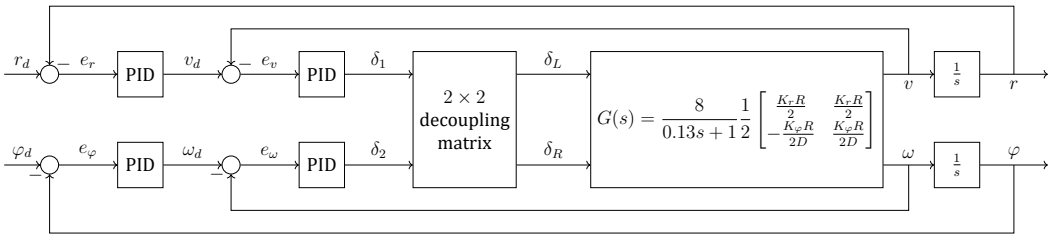


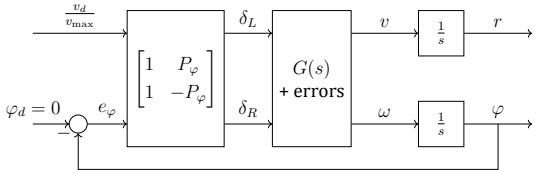
Fig. 9: *Attempted PID-based drive control structure*

4.4 Drive control logic

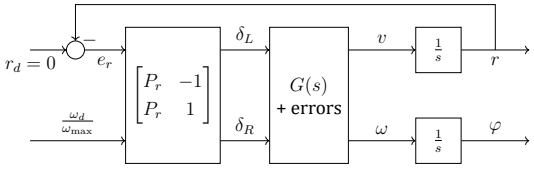
Driving controls are divided into rotations and translations. The driving logic implemented is a hybrid on-off controller for the target variable (r for translations, φ for rotations) and a proportional controller ($P_r = 0.01$, $P\varphi = 1$) for the unchanged variable (φ for translation, r for rotations). The local coordinates are reset after each movement command. The block diagram representation is shown in Fig. 10.

If a target distance or angle is specified, the rover rotates or translates to the target and brakes after reaching it, but this often led to overshoots due to inertia. Instead, the rover is made to brake when the target is within a certain range, given by a constant multiplied by the linear or angular velocity. It was found that, on a motor speed scale from 0 to 1, $P_\omega = 0.025$ and $P_v = 0.01$ are suitable constants for this purpose.

To move from point to point, the rover is first made to rotate through the principal angle required to orient the rover in the correct direction. The rover then translates until it reaches the point of interest.



(a) *Translation controller*



(b) *Rotation controller*

Fig. 10: *Movement controllers*

4.5 Timing and communication

The drive program is implemented as a library and runs on Core 1 of the ESP32 microcontroller as a high priority task interrupting the main loop for a brief interval of around $300\,\mu\text{s}$ at each sampling time. The latter was chosen to be $20\,\text{ms}$, a value large enough to avoid quantisation errors of the OFS when the speed is low, and small enough to avoid overflow when speed is high.

Vision

The vision system aims to detect aliens and alien buildings and measure their relative positions to the rover with their dimensions. The acquired data would be used to perform routing and mapping.

5.1 Alien Detection

Pre-Process Filter: Gaussian Filter

Since the Fourier-transformed of the Gaussian distribution has a low-pass filter characteristic, convolving an image with a Gaussian kernel is equivalent to multiplying the frequency spectrum of an image with a low-pass filter, suppressing high-frequency additive Gaussian noise.

Initially, a 3×3 kernel design was developed with a shifting RAM, which holds the information of three rows of pixels with one-pixel input and two-pixel output, allowing access to pixels from the previous two rows with a minimum delay. However, a more computational and memory-efficient way was found later and was taken as follows. Due to the jointly Gaussian distribution characteristic, its corresponding 2D kernel can be separated into two 1D vectors.

For 2D convolution, a total number of $(2K+1)^2$ computational operations is required, with the given filter size $2k+1$. In comparison, in the 1D case, the number of operations is kept as $2k+1$. Therefore, the total computation could be reduced to $2(2k+1)$ from $(2k+1)^2$. One valid method suggests discarding y-convolution and only recording the blurring effect according to horizontal pixels. It could primarily exploit the usage of logic elements, although some slight and durable distortions are introduced. An approximation is taken such that the convolution is done in integers to accelerate the calculation, as shown in Fig. 11a.

Pre-Process Filter: Median Filter

A median filter (Fig. 11b) is applied to reduce noise further. Each target pixel is replaced by the median values of its neighbouring entries within the predefined kernel window.

10	111	90	239	8	70	90	88
22	8	31	49	31	8	➡	29
60	33	190	220	219	200	23	9

(a) 1×5 *Gaussian filter, with results right shifted by 6 bits*

10	111	90	239	8	70	90	88
22	35	77	65	311	80	➡	29
60	33	190	220	219	200	23	9

(b) 1×5 *median filter, with median 77 assign as target*

Fig. 11: *1D Filter kernel*

HSV Conversion

Identifying specific colour under different light intensities based on RGB could be challenging as the three parameters are co-related to the colour luminance. The introduction of HSV vastly simplifies the tuning process since it isolates luminance from colour information.

The range of Value and Saturation of HSV is mapped to a [0-255] integer space to avoid floating-point arithmetic. Apart from that, a five-stage shifting register array is used to store five consecutive pixels, such that a target pixel would only be valid if it shares the same colour with its neighbourhood. This method reduces the noise effect.

Data Analyst

The pixels after HSV conversion would be analysed by the algorithm shown in Fig. 12 to deduce the bounding box of the alien. The general concept of the algorithm is to initially infer the possible region of the alien based on pixels in every row and confirm the region by increasing or decreasing the credibility of that estimated region with the accumulation of rows. A new estimated region would be considered when the region is no longer trustworthy. After examining all rows, the estimated region that meets the defined credibility requirement would be treated as the bounding box of the alien. This requirement is applied to avoid misidentifying the darkening colour at the bottom of light aliens, caused by the reflective light from the ground and the shadow of the alien. The accumulated dark colour at the bottom will not meet the set credibility requirement and, therefore, will not be wrongly considered as a new alien. Generally, this algorithm could achieve high accuracy while keeping the memory usage low for each corresponding colour.

Three rules are followed when deciding whether an inferred alien region selected from every row is valid

1. The number of pixels belonging to specified colour should be larger than a certain threshold to reduce the influence of noise in the environment.
2. The row deviations should conform to a predefined threshold which limits the median of the inferred region obtained in every row from largely deviating from the median of the estimated region concluded from the past N rows.
3. Regarding the boundary difference, the length of the inferred alien region is also limited to another certain threshold.

As shown in Fig 13a, the variable Trust represents the credibility of an alien located in the current estimated alien region. If the three rules above are met, this value is added by one, indicating that this region is more trustworthy. At the same time, the region would grow itself to adapt to the newly added segment. In contrast, this value is subtracted by one if any requirement is not met. When it reaches zero, the estimated region R would be reconsidered. At the end of every frame, the valid estimated region R with Trust meeting the set requirement is regarded as the bounding box of the alien. In distance measurement, the average bounding box of five continuous frames is computed to provide stable distance information.

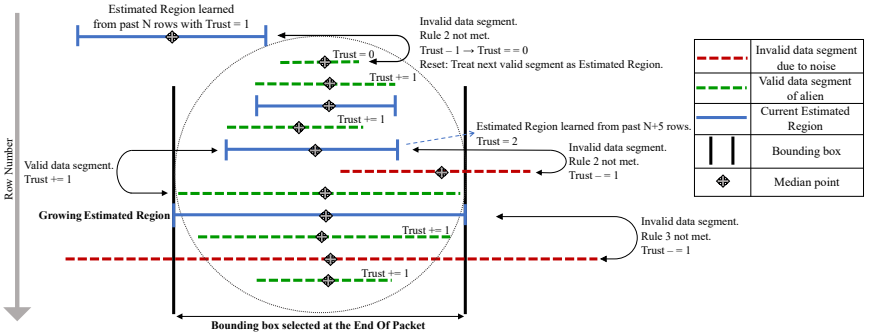
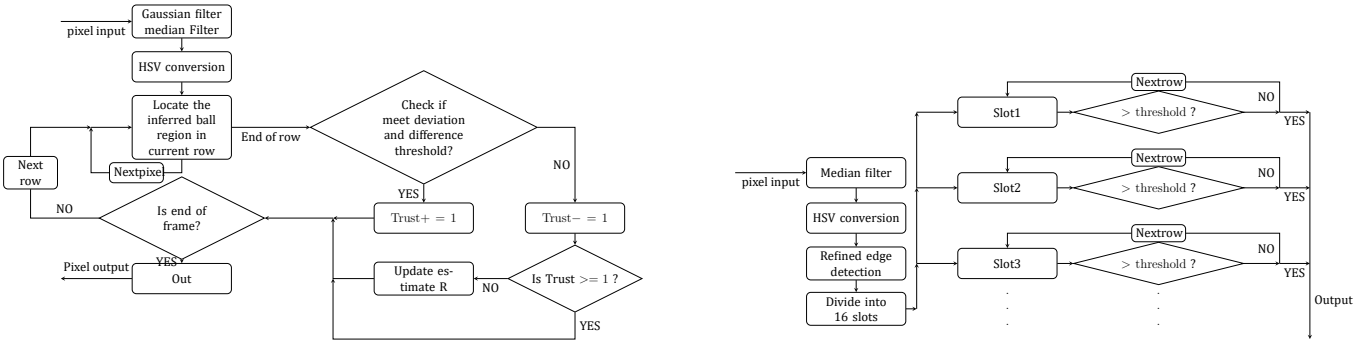


Fig. 12: *Alien Data Analyst Algorithm*



(a) *Alien Detection* **(b)** *Building Detection*

Fig. 13: *Detection Algorithm*

5.2 Building Detection

Pre-Process Filter

The median filter is embraced for building detection over other averaging filters, as it re-uses the image's existing pixels, retaining the building's edge information. Apart from that, it can remove stand-alone high intensity black and white pixels significantly. Hence, the detection is simplified as buildings are also in highly intense colour. In contrast, Gaussian filter is removed from the filter sets, as it gives rise to edge vanishing and phantom edges. [3]

Refined Edge Detection

Building detection is based on a different approach. Since the distance and dimension information is only relevant to the vertical edges of the building, the detection of horizontal edges is not considered, aiming to reduce computational complexity and unnecessary usage of logic elements. Initially, a 1D edge detection method was developed according to the luminosity in grey-scale space. However, the technique was shown to be non-robust regarding various lighting conditions where different levels of greyness appeared on white and black stripes that could be potentially misidentified as edges. To tackle this, a refined edge detection method based on the concept of increasing contrast was introduced (shown in Fig. 14), wherein the kernel

coefficients are unchanged but each pixel source is replaced by the possible classified HSV black(-1) and white(1) pixels. A total of 13 pixels are considered to differentiate an edge point between black and white stripes to compensate the "pixel deformation" in Appendix Fig. 27 observed by the experiment.

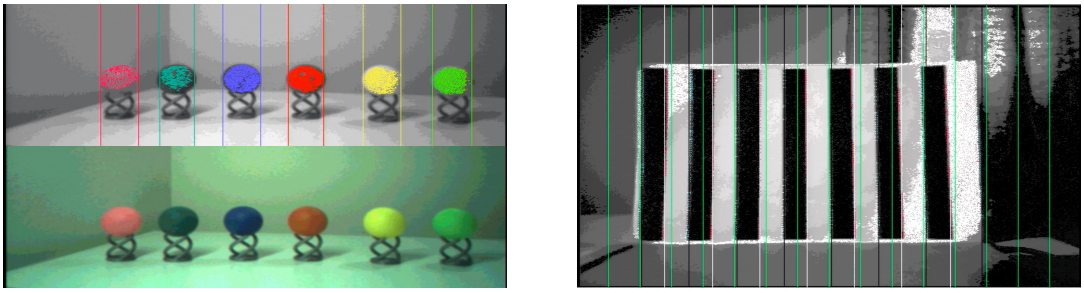
-1	-1	-1	-1	0	0	1	1	1	1	1	0	0	-1	-1
0	1	1	1	1	1	1	0	-1	-1	-1	-1	-1	-1	➡
-1	-1	0	0	0	0	0	1	-1	1	0	1	0	-1	-1

Fig. 14: 1×5 *Edge detection filter* $\{\text{white} = 1; \text{black} = -1; \text{others} = 0\}$

If the result is within a valid region, the target pixel is concluded as an edge point. The decision on the edge type white-to-black or black-to-white is made by comparing the number of black pixels and white pixels on the left and right of the kernel.

Clustering

The second step as shown in Fig. 13b is to find the edge position from the scattered possible edge points by utilizing the concept of divide and conquer. Every row of the frame is divided into 16 slots, 40 pixels/slot evenly to exploit the usage of the memory registers. The number of slots is carefully chosen by considering the focus length of the camera along with the previously knew strip width to ensure there could be at most one black-to-white and one white-to-black edge detected in every slot. Therefore, for each slot, only one decision is required for each edge type. The process divides the task into sub-tasks with the same input and output form suitable for applying the alien data analyst algorithm in Fig. 13a . Conquering the result from each slot would give rise to the final result.



(a) *Alien Detection result* **(b)** *Building Detection result*

Fig. 15: *Detection Flowchart*

Distance and Diameter measuring

To calculate the distance more accurately, the rover is instructed to rotate to the centre of the building, where the centre position is computed by comparing the number of edges in the left and right eight slots. The middle four slots with the least deformation are used to decide the width of the stripes. Many situations, such as misdetection of some edges, are well considered to provide a relatively accurate width in pixels. Apart from that, with the left and right bounds of the building identified by detecting the left-most and right-most edges, the diameter could be easily obtained in term of a constant factor C_{focus} by the following:

$$\frac{(\text{left}_{\text{most}} - \text{right}_{\text{most}}) \times C_{\text{focus}}}{\text{stripewidth}} \tag{2}$$

5.3 Solution to detect multiple objects within an image: The Lock and Block Mechanism

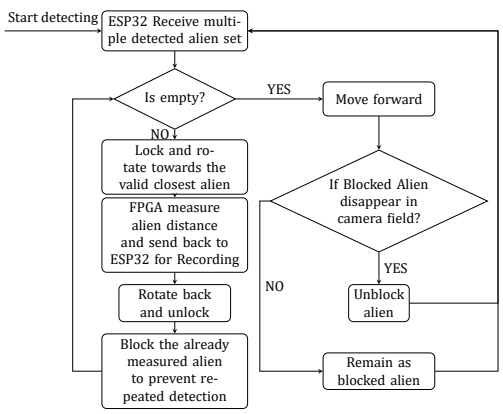


Fig. 16: *Lock and Block Mechanism*

Since the measurement is more accurate if the camera points toward the object, the rover is instructed to rotate towards the closest object based on rough distance measurement. During the rotation, new targets closer to the rover might enter the camera's visible region, which could lead the rover unexpectedly target another object. To tackle this, a locking mechanism is implemented such that FPGA will only focus on the locked targets. After the targeted alien is at the centre of the image, the rover would wait for 50 frames to collect the most reliable distance data, followed by rotating back and unlocking the target. Once a stable measurement is achieved on ESP32 side, an acknowledgement signal would be sent to FPGA, instructing it

to block the corresponding target and pop out the colour from the detected set. The remaining targets' information would then be transmitted to ESP32 by going through the steps above respectively until all the targets in the current frame are blocked, winding up the detection process for this node. The colour would be released from the blocking set when the corresponding alien becomes invisible to the rover. This enables the rover to detect the alien with the same colour multiple times, which is used in the confirmation part as discussed in the 'control' section.

5.4 Testing

Error and trial method is individually carried out for tuning targets' HSV and the decision threshold involved in the algorithm discussed above to ensure an accurate bounding is found for distance and dimension measurement. Calibration experiments are conducted under different lighting conditions to obtain a validated HSV parameter set. The percentage of valid distance within 50 frames is used as a metric to evaluate the performance of different filers.

6 Radar

The radar subsystem aims to locate underground power stations which are characterised by a reflective fan rotating at constant angular velocity.

6.1 Theory

The HB100 microwave radar module consists of a pair of two-element arrays of inset microstrip patch antennas. One acts as a sending antenna with frequency $10.5\,\mathrm{GHz}$, and the other acts as a receiving antenna. The spacing between the two array elements is around $1.5\,\mathrm{cm}$, that is, half a wavelength, hence the radiation pattern is expected to be more directed in one plane (say, the elevation) than the other (say, the azimuth).

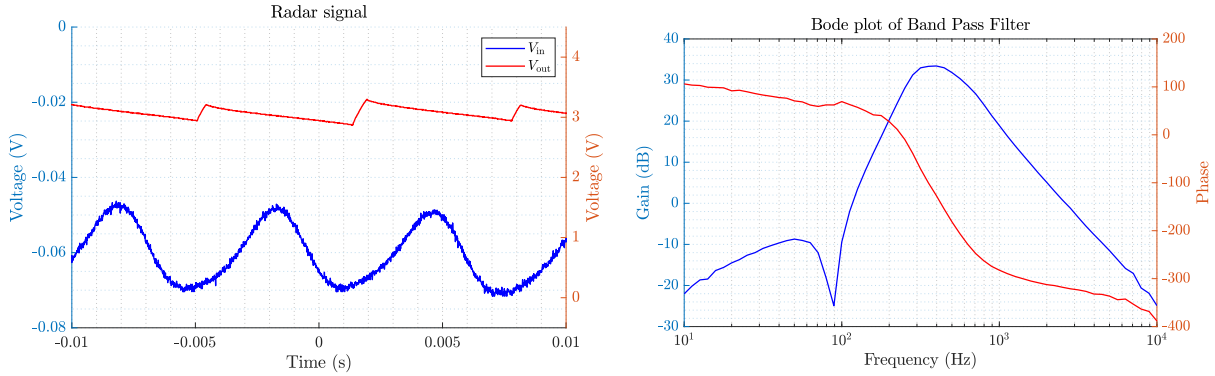
The radar module senses a target using the Doppler effect. In the non-relativistic limit ($v \ll c$), the frequency shift of the received signal is $\Delta f = \frac{v}{c} f_0$, where v is the target velocity, c is the speed of the wave, and f_0 is the frequency of the transmitted signal. The module has a radio-frequency (RF) circuit that gives an output intermediate frequency (IF) signal after mixing the received signal with the transmitted signal.

6.2 Experimentation and observations

From these observations, it was deduced that the following issues must be addressed:

1. The frequency signature of the fan was measured to be around $360\,\mathrm{Hz}$.
2. There is a noticeable amount of high frequency noise.
3. The amplitude of the signal goes up to $5\,\mathrm{mV}$ when placed near to the fan (Fig. 17a).
4. There is a DC offset in IF signal due to reflections from objects static in the rover frame (Fig. 17a).
5. Movement of the rover introduces very low frequency signals, since the surroundings are now moving in the rover frame.

To target issues 2, 4, and 5, the frequency separation between the clutter (low frequency), the noise (high frequency), and the target (intermediate frequency) can be exploited and the target signature can be isolated with a band-pass filter around $360\,\mathrm{Hz}$. To target issue 3, amplification is needed.



(a) *Radar signal* **(b)** *Bode plot of Band Pass Filter*

Fig. 17: *Experimentation with Radar (left) and Frequency response of band pass filter (right)*

6.3 Circuit Design

The complete circuit diagram is as shown in Fig. 18. The input of the signal, V_{in} is from the radar module and the output, V_{out} will be a DC signal. The design consists of 5 stages:

- 1st (Non-inverting amplifier): Remove negative DC offset, incur new DC offset at $2.5\,\text{V}$ to give enough room for the amplified signal and amplify with a gain of 500 for easier detection.
- 2nd & 3rd (Band pass filter): Isolate the target signature signal around $400\,\text{Hz}$, attenuate low frequency clutter and high frequency noise. Has a bandwidth of $500\,\text{Hz}$ to accommodate for the fact that the IF signal is not a pure harmonic, since the fan has multiple blades.
- 4th (Inverting DC-removing amplifier with half-wave rectification): Remove DC offset, amplify with a gain of -8 and half-wave rectify the AC signal.
- 5th (Peak detection): Smoothen the rectified signal into a constant DC signal.

V_{out} is then fed to an analogue-to-digital conversion (ADC) pin on ESP32 for further data processing. This design makes use of the full dynamic range of the ADC, that is, $0 - 3.3\,\text{V}$.

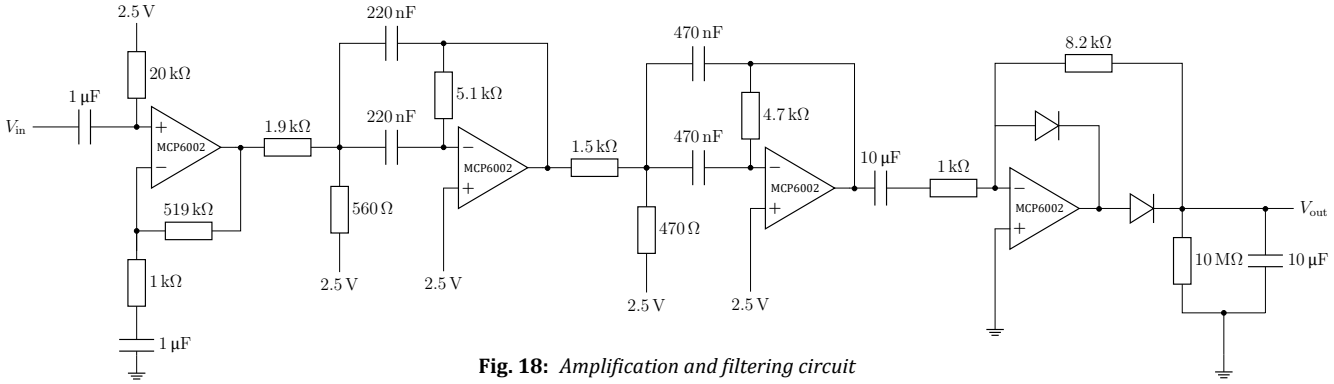


Fig. 18: *Amplification and filtering circuit*

6.4 Radar position and angle, and its effect on V_{out}

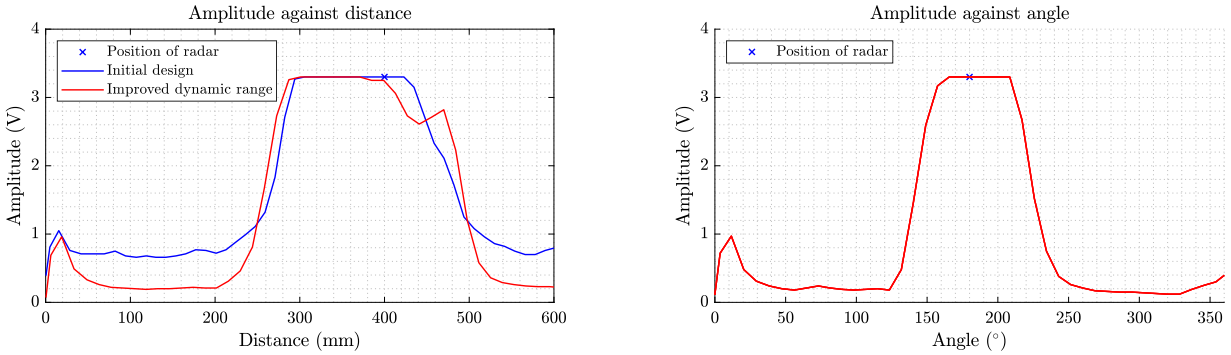


Fig. 19: *Effect of distance and angle on the amplitude of* Vout

The radar module is installed below the rover at about $45\,^\circ$ from the ground, hence the amplitude reaches $3.3\,\text{V}$ for a longer distance (Fig. 19) when the rover approaches the target. It is also observed that the improvised design has a better dynamic range, compared to a previous design where an offset of 0.5V remained. In terms of angle, the amplitude increases as the rover turns towards the target. The fact that the elevation pattern is more directed manifests itself in the narrower detection band in angle sweep.

At start up, there is a peak in amplitude at about $1\,\text{V}$. This is suspected to be due to the electrical noise from power rails because both experiments involve 2 completely different motion (translation & rotation).

6.5 Microcontroller logic, server and database communication

The microcontroller is programmed to sample the ADC pin at $50\,\text{Hz}$ and calculate the moving average of 10 samples (using vectors in C++) to reduce the effects of voltage ripple. A threshold of $0.2\,\text{V}$ is then set, for which it is considered that a fan is approaching if value $> 0.2\,\text{V}$. If a fan is approaching, the voltage values are sent to the server as an indication of possible distance and orientation. Due to the distributed size of the fan and uncertainties from the sensor, it is mapped to a region on the map instead of a point.

Command

The command system aims to enable the interaction between the Mars rover and user endpoints. As shown in Fig. 20, the command system contains four main parts - ESP32, database server, web server, and web client. The web server and the database server are separated to perform their designated jobs, simulating real-life scenarios. The database server focuses on receiving, storing, and redistributing rover's data, while the web server processes users' requests. This arrangement reduces processing loads for each server.

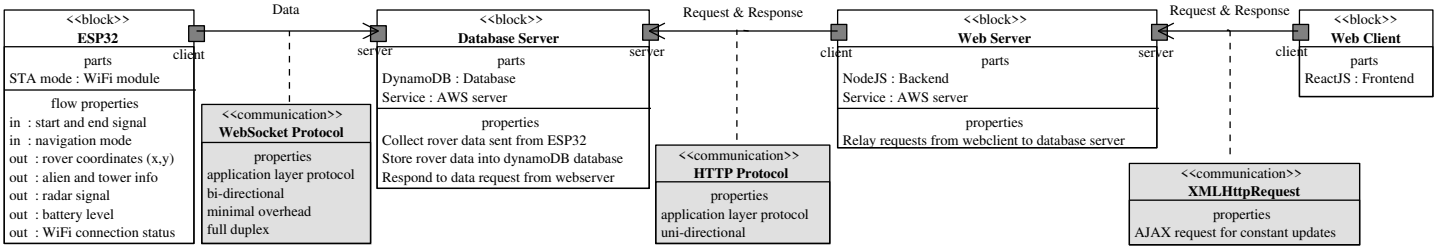
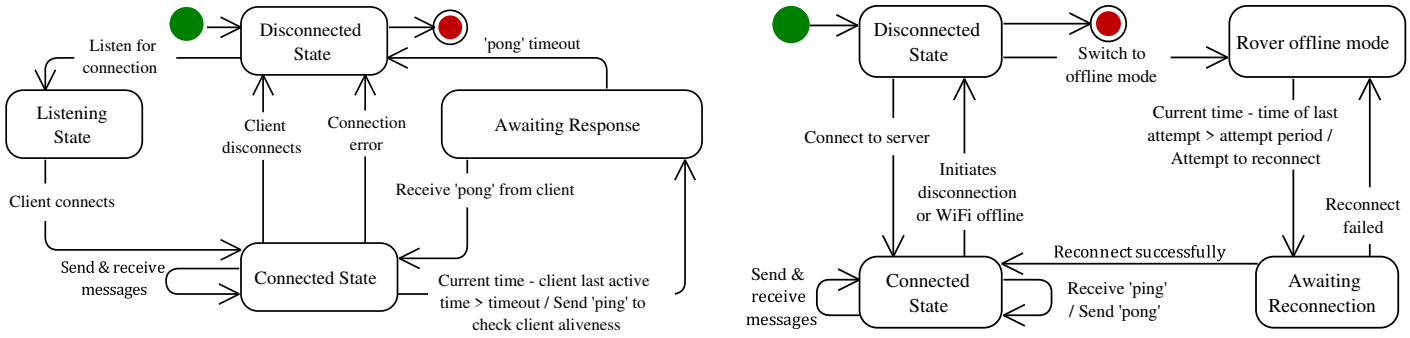


Fig. 20: *Block definition diagram for structural design of command system*

7.1 ESP32 & Database Server communication

WebSocket application-layer protocol is utilised due to its full-duplex and bi-directional characteristics, which enable both the database server and ESP32 to initiate a data transmission process. Initially, TCP transport-layer protocol was used, but after some exploration, WebSocket is chosen because it provides a more efficient implementation. The recipient of a message for WebSocket protocol is event-driven (i.e. a message handler routine is registered), whereas the TCP socket uses byte streams for transmission. Hence the message might be divided at the receiving end. As a result, no further reassembling is needed for WebSocket protocol, but extra framing is required to reassemble bytes into a message if TCP transport-layer protocol is used [4]. From measurements, the average Round Trip Time (RTT) for a byte to be transmitted between ESP32 and the AWS server is $82.3\,\mathrm{ms}$, which is slightly longer than the $80.7\,\mathrm{ms}$ RTT for TCP protocol within a tolerable range. However, WebSocket protocol provides a more efficient implementation.

Due to the long distance between the rover (on Mars) and the database server (on Earth), data is converted to short code for traffic reduction. WiFi signal is presumably unstable on Mars. Hence, a robust scheme is developed to deal with the rover losing connection with the database server, as shown in Fig. 21 below.



(a) *Database server*

Fig. 21: *State machine diagram*

(b) *ESP32 client*

A NoSQL database (DynamoDB) is utilised to handle large volumes of data at high speed. The stored path and exploration details for each navigation can be retrieved and presented visually on the website.

7.2 Database server & Web server communication

HTTP protocol is used for the communication process with the web server acting as 'client' and database server acting as 'server'. Upon receiving the web server's requests, the database server will query the database and assemble its response in .JSON format to facilitate easy parsing at the web server side.

7.3 Web server & Web client communication

AJAX is utilised to allow exchanging messages with the web server asynchronously behind the scene and update parts of the web page dynamically.

The control system is mainly in charge of exchanging data from other relevant subsystems and deciding the subsequent movement of the rover based on various algorithms.

8.1 Exploration Mode

Exploration Procedure

The total arena is separated into 11×17 nodes to aid the path-finding algorithm, which has been designed to dynamically change the route to avoid detected obstacles. In order to detect all possible targets, the full coverage of the map needs to be ensured. The exploration algorithm is shown in Fig. 22. At each node, an obstacle detection procedure is carried out as shown in the flowchart below (Fig. 23). [5]

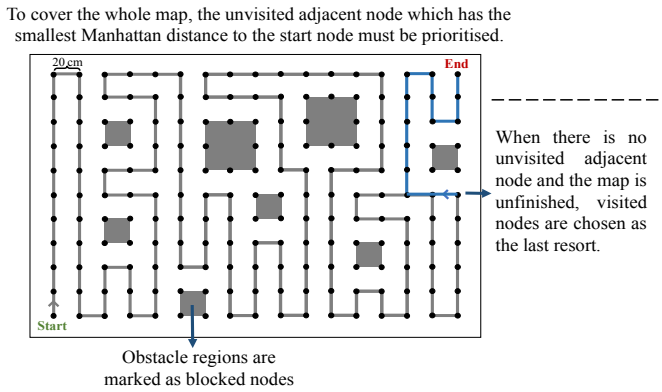


Fig. 22: *Exploration Algorithm*

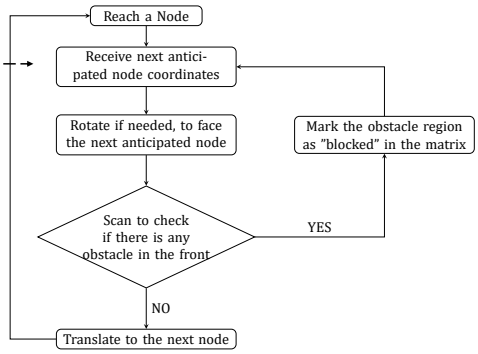


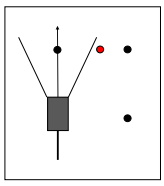
Fig. 23: *Alien Detection*

Confirmation

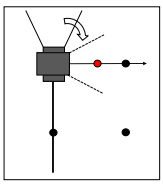
The designed exploration algorithm ensures each alien or tower to be detected at least twice from different positions, avoiding mistaken detection. When detecting a specific alien for the second time, ESP32 will compare the result with the previous measurement. If the two measurements are close, the rover will average the two results. Meanwhile, on the server-side, the alien displayed on the website will adjust its position. Conversely, if a significant deviation was obtained from the two measurements, this alien will be remembered as "uncertain" and temporarily erased from the server-side map. After exploring the whole arena, the rover will search for the "uncertain" alien and remeasure its position to ensure accurate mapping.

Obstacle Avoidance

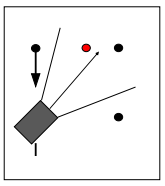
Three ToF (Time of Flight) sensors are introduced such that the rover is aware of its neighborhood which is within the blind spot of the camera. If any obstacle is detected within this region, an interrupt will be triggered, instructing the rover back to its previous node position while aligning its camera with the anticipated node. The camera would then reexamine the node and update the map if needed. After that, the rover would return to its initial position before the interrupt and give the control back to the main sequence.



(a) *At previous Node, alien not detected*



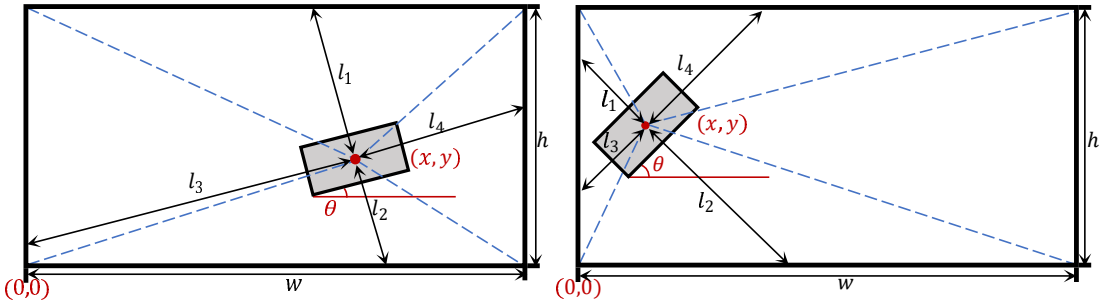
(b) *Facing the anticipated node, alien detected by Tof*



(c) *Return Back and reexamine that node*

Error Correction

Four HC-SR04 ultrasonic devices are placed on the rover, facing four directions perpendicular to each other. Due to the surrounding noise, distance data collected by the ultrasonic devices is unstable. Hence, a low pass filter is added for simple noise suppression purposes. At the later phase of the exploration, the optical flow sensor, which has accumulated error, gradually becomes inaccurate. In comparison, the ultrasonic device, which has constant error, is thus used for error correction. When the rover reaches different locations of the arena, different equations need to be applied for position coordinate calculation. Fig. 25 shows two scenarios, where l_1, l_2, l_3, l_4 are distances measured by four ultrasonic devices, (x, y) represents current rover coordinates and θ is the tilting angle of the rover.



(a) *Example situation 1* **(b)** *Example situation 2*

Fig. 25: *Example situations for distance measured by the ultrasonic device*

The general equations are summarised as:

$$\begin{cases} x = l_4\cos(\theta), & \theta < \tan^{-1}(\frac{y}{x}) \\ y = l_4\sin(\theta), & \text{otherwise} \end{cases} \qquad \begin{cases} y = l_2\cos(\theta), & \theta < \tan^{-1}(\frac{w-x}{y}) \\ x = w - l_2\sin(\theta), & \text{otherwise} \end{cases}$$

$$\begin{cases} x = x - l_3\cos(\theta), & \theta < \tan^{-1}(\frac{h-y}{w-x}) \\ y = h - l_3\sin(\theta), & \text{otherwise} \end{cases} \qquad \begin{cases} y = h - l_1\cos(\theta), & \theta < \tan^{-1}(\frac{x}{h-y}) \\ x = l_1\sin(\theta), & \text{otherwise} \end{cases}$$

8.2 Subsystem communication

ESP32 communicates with different components using different communication protocols.

Subsystems	Usage	Protocol	Characteristics
FPGA ⇔ ESP32	Transmit vision data	SPI	Faster transmission speed, suitable for transmitting large amount of data
Arduino Nano Every ⇔ ESP32	Transmit ultrasound and ToF sensors measured data	UART	Fast enough for transmitting small amount of data
Database server ⇔ ESP32	Communicate with the database server	WiFi	Wireless transmission with security; Large connection range

Table 4: *Communication Protocols for ESP32*

8.3 Additional Modes

Besides the basic exploration modes, additional functions are also implemented.

1. The rover can move to a destination or an alien specified by the endpoint user. A* algorithm is used for shortest path calculation. [6]
2. The rover must return to the origin when battery level is low. ESP32 would constantly compute the shortest path to the origin using A* algorithm and decide whether returning is possible and necessary based on the current battery level.
3. Remote control is implemented. The command subsystem is directly linked with the drive subsystem for this function.

9 Integration and Testing

The testing methodology for each individual subsystems are described in the above sections. The integration testing process starts from connecting each individual subsystem to the control subsystem.

- **Vision + Control**: A debugging interface between ESP32 and FPGA is implemented which allows users to pick up to ten 16-bit variables from FPGA, to be displayed on the ESP32 console at the start of each map node.
- **Command + Control**: A mock ESP32 client program is created which constantly sends simulated rover data to the database server for web app rendering. WiFi connection instability of ESP32 is simulated by manually closing and reopening WiFi hot spot. it is tested that an ESP32 disconnection can be detected and displayed on the web app.

After making sure each subsystem can communicate with ESP32, inter-modular testings involving three

related subsystems are conducted. In this way, defects in the interaction between the subsystems can be exposed, making fault localisation easier.

- **Vision + Control + Drive**: Various placements of balls and towers, including marginal situations like many aliens placed together at the start, were used to test the behaviour of detecting and measuring the distance between the rover and the obstacles. Besides, a few complete explorations on Mars Arena were conducted.
- **Command + Control + Drive**: The testing procedure was designed in such a way that after a user clicks the 'start' button on the web app and ESP32 receives the 'start' signal, the rover will perform a sequence of simple predefined movements. This process checks the accuracy and synchronisation of the rover's path display on the web app. The remote control function was also tested at this stage.

Then, the four main subsystems (i.e. command, control, vision and drive) were integrated and tested before combining radar and energy subsystems to conduct the overall system test.

10 Evaluation and Future Work

The outcome of each subsystem is evaluated below and possible improvements are suggested:

- **Energy**: The overall design is fairly robust because many factors such as the battery's BMS, MPPT effectiveness and the system's efficiency were taken into considerations. In future, power from the Sun can be maximised further using a pan-and-tilt module. Moreover, an autonomously-charging mechanical system could be implemented in order for the rover to charge itself on Mars.
- **Drive**: Pure rotational and translational motion are reliable on most surfaces. Possible extensions to implement curved motion could be considered wherever navigation algorithms require such.
- **Radar**: The full dynamic range of the ADC pins are utilised in the radar circuit. Further improvements could be to use a digital potentiometer to increase resolution so a gradient radar map can be visualised on the website.
- **Vision**: Neural network has been considered for more accurate obstacle recognition. However, even with the the fastest transmission protocol SPI, the time needed to transmit a frame data from FPGA to ESP32 is lower-bounded by 80s. This leaves no option but to perform the data within FPGA. Transmission speed barrier needs to be overcome since it is the prerequisite for applying neural network.
- **Command**: All requirements are met but could be improved by adding Secure Socket Layer (SSL) and Transport Layer Security (TLS) to establish authenticated and encrypted links.
- **Control**: The current driving system is overly reliant on the optical flow sensor, which is heavily affected by external conditions (i.e. surface and lighting). Kalman Filter Algorithm can be used for continuous error correction during the exploration. Rover's coordinates collected from the optical flow sensor and ultrasonic devices can be used as measurements. A state space transition model can be developed by combining Eq. 1 and time-based equations describing the rover trajectory, where (x, y) is rover's coordinate and θ is rover's angle:

$$\begin{aligned} x_t &= x_{t-1} + \dot{r}_{t-1}\cos(\theta_{t-1} + \dot{\varphi}_{t-1}\Delta t) \\ y_t &= x_{t-1} + \dot{r}_{t-1}\sin(\theta_{t-1} + \dot{\varphi}_{t-1}\Delta t) \\ \theta_t &= \theta_{t-1} + \dot{\varphi}_{t-1}\Delta t \end{aligned} \tag{3}$$

  The nonlinear system can be linearised by Taylor Expansion, and then applied in the Kalman filter algorithm to obtain the best estimate of the rover's location and orientation.

To meet the non-functional requirement, a suitable balance has been chosen between exploration time and accurate obstacle detection. The rover is able to complete the exploration within a reasonable amount of time. The website interface is easy to use and the communication latency is below the required $200\,\mathrm{ms}$.
All functional requirements outlined in the requirement diagram (Fig. 1) are met. The rover can successfully detect and avoid obstacles, charge using solar energy, explore the arena autonomously and communicate with the server.

References

[1] R. A. Messenger and J. Ventre, *Photovoltaic Systems Engineering*. Boca Raton: CRC Press, 2010.

[2] T. B. Reddy, *Linden's Handbook of Batteries*. New York: McGraw-Hill, 2019.

[3] G. Deng and L. Cahill, "An adaptive Gaussian filter for noise reduction and edge detection," in *Proc. of Nuclear Science Symposium and Medical Imaging Conference*, 1993, pp. 0–5.

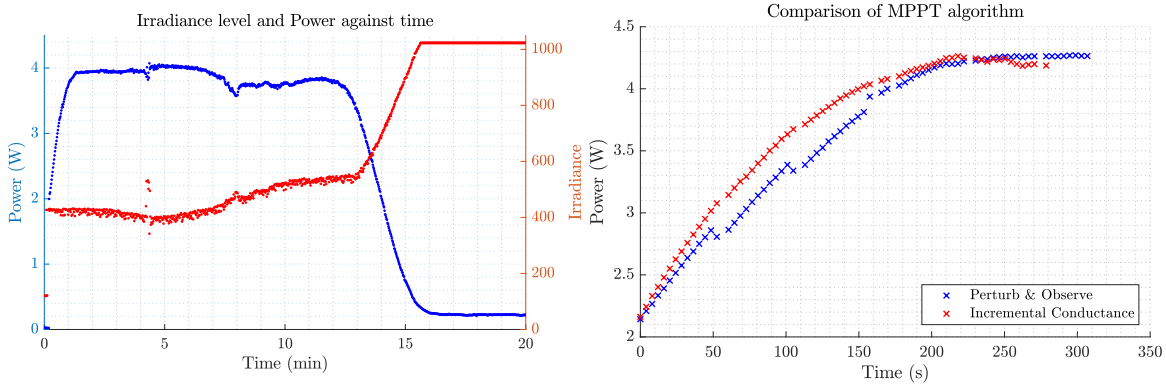
[4] D. Skvorc, M. Horvat, and S. Srbljic, "Performance evaluation of websocket protocol for implementation of full-duplex web streams," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2014, pp. 1003–1008.

[5] M. A. Yakoubi and M. T. Laskri, "The path planning of cleaner robot for coverage region using genetic algorithms," *Journal of Innovation in Digital Ecosystems*, vol. 3, no. 1, pp. 37–43, 2016, special issue on Pattern Analysis and Intelligent Systems – With revised selected papers of the PAIS conference.

[6] K. Magzhan and H. M. Jani, "A review and evaluations of shortest path algorithms," *International journal of scientific & technology research*, vol. 2, no. 6, pp. 99–104, 2013.

Appendix

11.1 Energy



(a) *Irradiance level and Power over time*

(b) *Comparison of P&O and IC*

Fig. 26: *MPPT*

		2S2P	4S	2S2P	4S
		From 3.7V to (V)	From 8.6V to (V)	From 34.6mA to (mA)	From 16.8mA to (mA)
Dark	Light	3.6	8.3	34.8	16.2
Dark	Light				
Light	Dark	3.7	8.1	20.5	15.8
Light	Dark				
Light	Light	2.2	8.3	19.2	16.15
Dark	Dark				
Dark	Dark	2	8.2	18.5	15.9
Light	Light				
Dark	Dark	2.2	3.2	20.5	6.2
Dark	Light				
Light	Dark	2.2	3.2	20.5	6.2
Dark	Dark				

Table 5: *Effect of using bypass capacitors on partial shading*

11.2 Vision

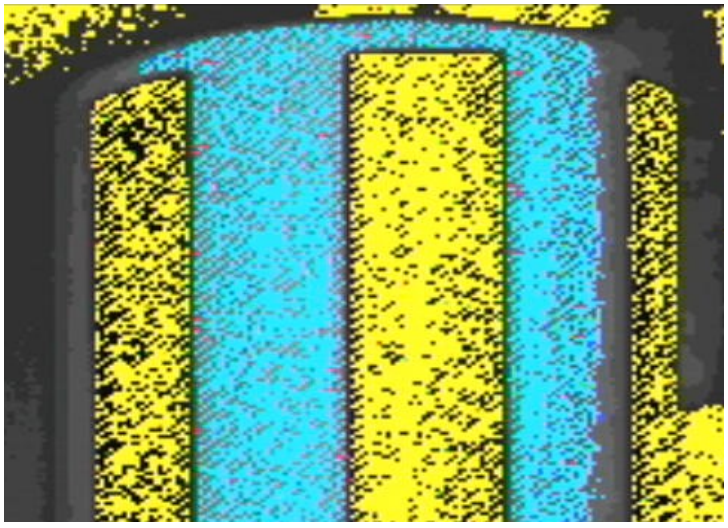


Fig. 27: *pixel distorted around the edges in a slash shape*

SourceCode, source code for clustering

```
if(x is within slot 1) begin // pure or
    if( blackToWhite_detected ) begin
        if(slot_1_count_bw == 0) begin
```

```
            slot_1_bw_tmp <= x;
            slot_1_count_bw <= 1;
            slot_1_blackToWhite <= 0;
        end
    else if(slot_1_count_bw <= bwb_threshold) begin
            if(((x >= slot_1_bw_tmp) && (x-slot_1_bw_tmp <= 5)
            )
            ||((x < slot_1_bw_tmp) && (slot_1_bw_tmp - x <= 5)))begin
                slot_1_count_bw <= slot_1_count_bw + 3;
            end
            else slot_1_count_bw <= slot_1_count_bw - 1;
        end
        else if((slot_1_count_bw > bwb_threshold) && (slot_1_blackToWhite !=
           slot_1_bw_tmp)) begin
            slot_1_blackToWhite <= slot_1_bw_tmp;
            count_b_w <= count_b_w + 1;
        end
    end
    else if(whiteToBlack_detected) begin
        if(slot_1_count_wb == 0) begin
            slot_1_wb_tmp <= x;
            slot_1_count_wb <= 1;
            slot_1_whiteToBlack <= 0;
        end
        else if(slot_1_count_wb <= bwb_threshold) begin
            if(((x >= slot_1_wb_tmp) && (x-slot_1_wb_tmp <= 5) )
            ||((x < slot_1_wb_tmp) && (slot_1_wb_tmp - x <= 5)))begin
                slot_1_count_wb <= slot_1_count_wb + 3;
            end
            else slot_1_count_wb <= slot_1_count_wb - 1;
        end
        else if((slot_1_count_wb > bwb_threshold) && (slot_1_whiteToBlack !=
           slot_1_wb_tmp)) begin
            slot_1_whiteToBlack <= slot_1_wb_tmp;
            count_w_b <= count_w_b + 1;
        end
    end
end
.
.
.
```

Alien Data Analyst

```
Pink alien example
//------------------------------------------------------------------------
assign mid_deviation_p =  ((estimatated_region_end_p + estimatated_region_start_p)
    > (max_start_edge_x_position_p + max_end_edge_x_position_p)) ?
((estimatated_region_end_p + estimatated_region_start_p) - (
   max_start_edge_x_position_p + max_end_edge_x_position_p)): ((
   max_start_edge_x_position_p + max_end_edge_x_position_p) - (
   estimatated_region_end_p + estimatated_region_start_p ));
assign difference_p = max_end_edge_x_position_p - max_start_edge_x_position_p;
//------------------------------------------------------------------------
//Pink
if(count_p > count_threshold) begin
 if(estimated_val_p == 0)begin
  estimatated_region_start_p <= max_start_edge_x_position_p;
  estimatated_region_end_p <= max_end_edge_x_position_p;
```

```
  //reset
  x_min_p <= IMAGE_W-11'h1;
  x_max_p <= 0;
  estimated_val_p <= 1;
 end
 else begin
  if(mid_deviation_p > horizontal_edge_region_threshold)begin
   estimated_val_p <= estimated_val_p - 1;
  end
  else begin
  if(difference_p < difference_threshold )begin
   estimated_val_p <= estimated_val_p + 1;
   if(x_min_p > max_start_edge_x_position_p) begin
    x_min_p <= max_start_edge_x_position_p;
   end
   if(x_max_p < max_end_edge_x_position_p) begin
    x_max_p <= max_end_edge_x_position_p;
   end
  end
 end
end
end
```

11.3 **Integration**

Aliens Position accuracy	Alien Tower Position accuracy	Alien Tower Diameter
W/Without Avg Filter	$90\% \pm 5\%$	$78\% \pm 10\%$
w/Without Gaussian Filter	$88\% \pm 8\%$	$75\% \pm 11\%$

Table 6: *filter accuracy evaluation*