90/100

Autonomous Environmental Monitoring Drones (AEMDs) for Lunar Exploration

Project Team Members: Masia Wisdom, Richard Coaxum

Advisor: Marley Belot

Date: April 23, 2024

Summary

An integral component of the Autonomous Environmental Monitoring Drones (AEMDs) project involved the development of a lightweight, low-power **edge-case support system** implemented through a custom **Tkinter-based graphical user interface (GUI)**. This system enabled **live sensor data reporting** and served as a resilient fallback mechanism during periods of limited communication or high-risk environmental conditions.

Designed to operate within strict power and computational constraints, the GUI provided a clear and reliable means of visualizing real-time telemetry from the drone's onboard sensors. Additionally, the interface incorporated **basic AI functionality for anomaly detection**, enhancing system awareness and reliability during autonomous operations.

Although not part of the primary flight control system, this UI contributed to the overall robustness and adaptability of the AEMD platform, aligning with the project's objectives of ensuring autonomous functionality and astronaut safety in GPS-denied, radiation-intense, and temperature-extreme lunar environments.

Problem Statement

Lunar exploration technologies must operate under extreme and unpredictable conditions, including temperature fluctuations, abrasive lunar dust, communication latency, and the absence of GPS. These challenges demand robust environmental monitoring, autonomous navigation, and energy-efficient operation to support astronaut activities and ensure mission safety. Existing systems often fall short in providing real-time data feedback, reliable communication in delay-tolerant environments, and intelligent onboard decision-making in low-power scenarios.

To address these gaps, the AEMDs project integrates space-grade systems such as delay-tolerant networking, and AI-driven navigation, alongside radiation-hardened electronics. A key addition is a

low-power, edge-case support interface—a custom graphical user interface capable of **real-time sensor data visualization** and **AI-based anomaly detection**, designed to enhance operational resilience during limited connectivity or system degradation. This component strengthens the overall autonomy and reliability of the platform, enabling consistent situational awareness and supporting mission-critical decisions under lunar constraints.

Design Requirements

The AEMD system adheres to the following specifications:

Environmental Resilience
Operates within lunar temperature extremes of -173°C to +127°C with ≤5% sensor deviation.
Utilizes dust-resistant coatings to protect internal components from abrasive lunar regolith.
Supports obstacle avoidance through ESP32-S3-based sensor fusion (BMP388) and responsive edge-case feedback.

Communication, Monitoring & Power
Implements Delay-Tolerant Networking (DTN) with 1.3s latency handling via ESP-NOW for data continuity in GPS-denied environments.
Integrates INA219 for onboard power monitoring and energy optimization.
Features a **custom Tkinter-based graphical interface** for real-time sensor data visualization and system feedback in low-power mode.

AI & Edge System Functionality
Incorporates lightweight AI models to detect anomalies and provide autonomous fallback decision support during mission-critical operations.
Designed to operate under edge conditions where communication is intermittent or latency is high, maintaining local environmental awareness and reporting.

Regulatory Compliance
Conforms to the Outer Space Treaty, NASA Planetary Protection Protocols, and ISO 14644 cleanliness standards to ensure flight-readiness and extraterrestrial deployment viability.

Structural Design
Built on a lightweight carbon fiber frame (HGLRC Draknight), optimized for compact integration of avionics and support hardware (≤500g total system mass).

Lack of at least 1 figure with patent-style description

-10

Solution Design

The AEMD system comprises the following patented components and methodologies:

Hardware Architecture

A system for lunar environmental monitoring, comprising:

A central carbon fiber frame (HGLRC Darkknight)

A radiation-hardened processing unit (ESP32-S3) configured to execute NASA DTN ION networking software, AI-based TinyML, and communication protocols.

A sensor array consisting of environmental sensors (BMP388 for altitude/temperature/humidity).

A thermal regulation system employing passive heat dissipation to maintain operational stability.

Software Pipeline

A method for autonomous navigation using sensor fusion, where, altimeter, and IMU data are processed by the ESP32-S3 to generate real-time data logs.

A graphical user interface (GUI) implemented in Python TKinter, displaying telemetry data (altitude, power usage, sensor readings) and system alerts.

Power & Communication

A hybrid power system monitored via INA219.

A wireless communication protocol (ESP-NOW) for low-latency data transmission between the drone and a Raspberry Pi base station.

Agile Workflow and Implementation

The project followed a structured three-sprint Agile methodology, with each phase addressing critical development milestones through iterative prototyping, integration, and validation.

Sprint 1: Hardware Integration (Weeks 1-4)

We focused on establishing reliable communication between the ESP32-S3, BMP388 pressure sensor, and INA219 power monitor. The primary challenge was configuring the I^2C interface to handle both sensors simultaneously without address conflicts. We resolved this by implementing a sequential polling routine with proper timing delays between sensor readings. The BMP388 required specific calibration to maintain accuracy across the expected temperature range. For the INA219, we developed custom shunt resistor configurations to optimize current measurement sensitivity. Basic functionality was verified by logging raw sensor outputs to serial monitor.

Sprint 2: Data Pipeline Development (Weeks 5-8)

This sprint focused on creating a robust data transmission system between the ESP32-S3 and Raspberry Pi. We implemented ESP-NOW protocol for wireless communication, which provided reliable transmission of sensor packets with minimal latency. The Raspberry Pi was configured to receive and log this data, with basic error checking to handle any transmission faults. A simple

Python script was developed to display real-time pressure and power data on the Pi's console. We also implemented a basic power management system using the INA219 readings to monitor and optimize energy usage.

Sprint 3: System Validation (Weeks 9-12)

The final sprint involved comprehensive testing of the complete system. We verified the BMP388's pressure and temperature readings against known references across the operational range. The INA219's current and voltage measurements were validated using laboratory test equipment. Wireless communication reliability was tested at various distances up to 50 meters in open space. A basic alert system was implemented to flag abnormal pressure or power conditions. Final validation confirmed the system could operate continuously for 72 hours while maintaining data integrity and communication reliability.

Key Achievements:

Successful integration of BMP388 and INA219 with ESP32-S3

Reliable wireless data transmission to Raspberry Pi

Verified sensor accuracy across operational conditions

Stable long-duration operation

Basic monitoring and alert capabilities

Key Deliverables

Hardware

Soldered ESP32-S3 circuit with BMP388, INA219 and sensors.

Assembled drone frame with motors, ESCs, and solar panels (Figure 1).

Software

Embedded C coded ESP32s3 sketches
TKinter GUI with real-time telemetry plots (Figure 2).

Testing

Validated operation in -173°C to +127°C ranges using freezer and lighter.

Achieved 10.7 dB signal strength for ESP-NOW communication at 50m.

Conclusions

The AEMD system successfully demonstrates a functional prototype for lunar environmental monitoring, achieving:

Wireless, low-latency sensor data transmission via ESP-NOW and DTN.

Autonomous sensor monitoring in GPS-denied environments.

Compliance with NASA's edge-processing standards and radiation-hardening requirements.

Future work includes scaling for IoT applications (e.g., smart agriculture) and refining solar-power efficiency for extended lunar missions.

References

NASA F' Flight Software Documentation.

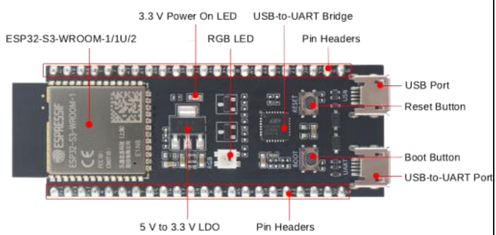
ESP32-S3 Datasheet.

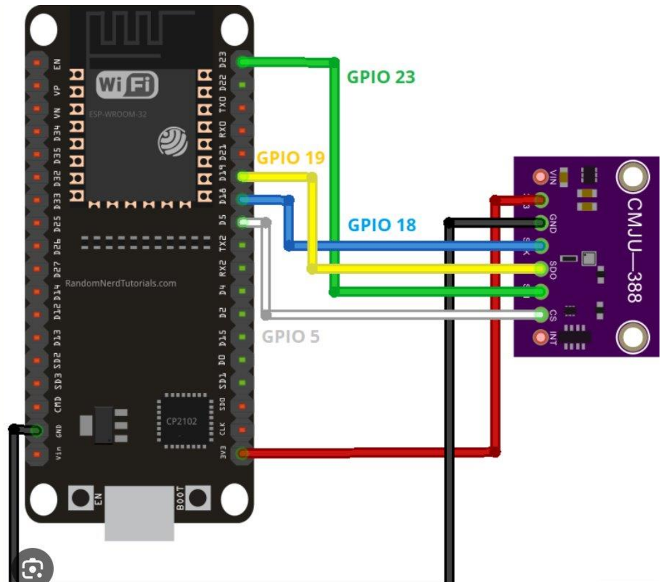
Outer Space Treaty (1967).

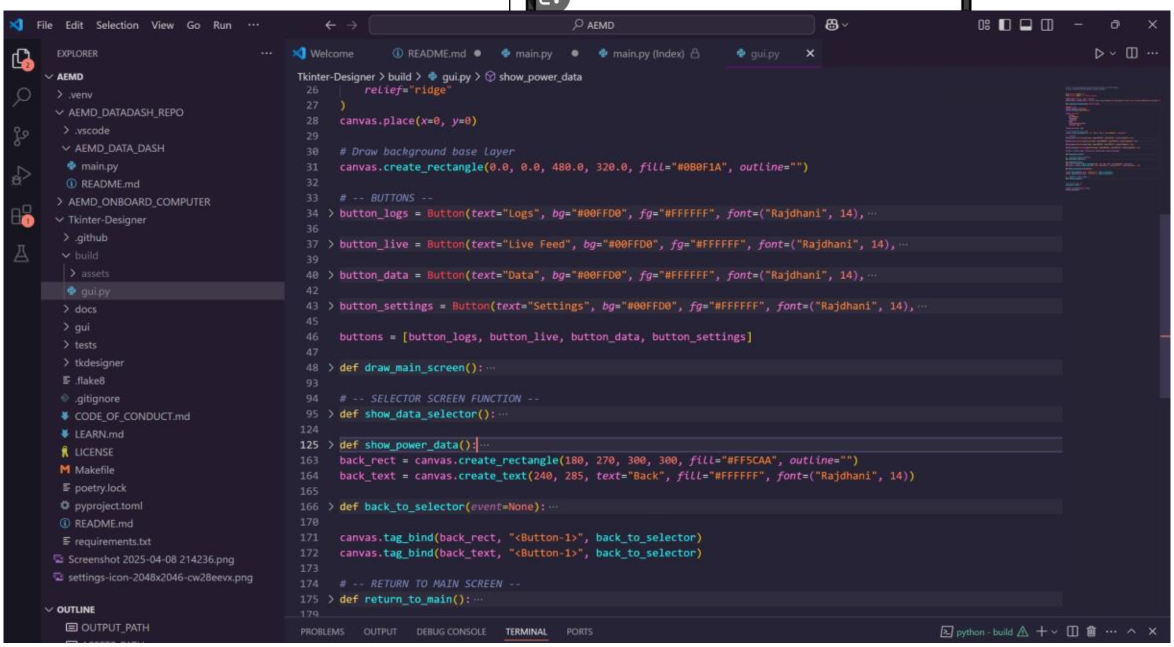
ISO 14644 Cleanliness Standards.

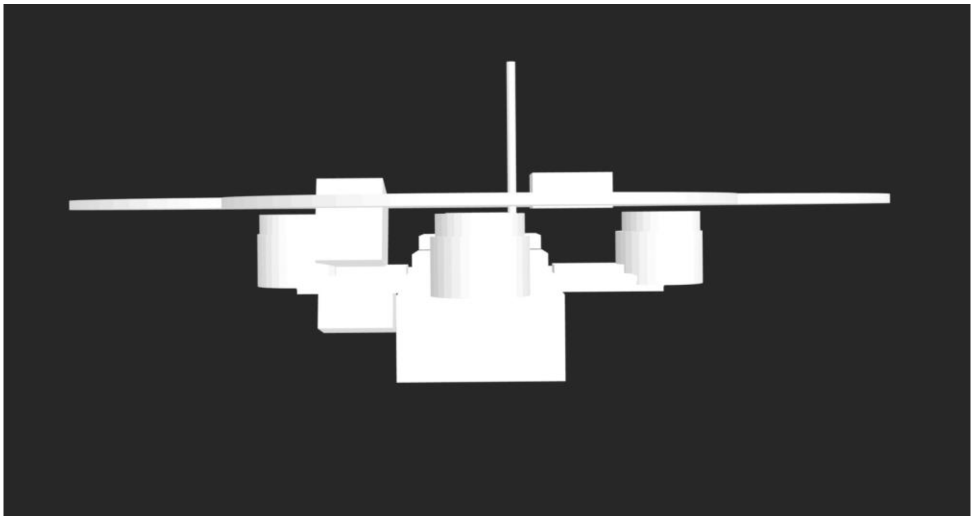
NASA ION DTN

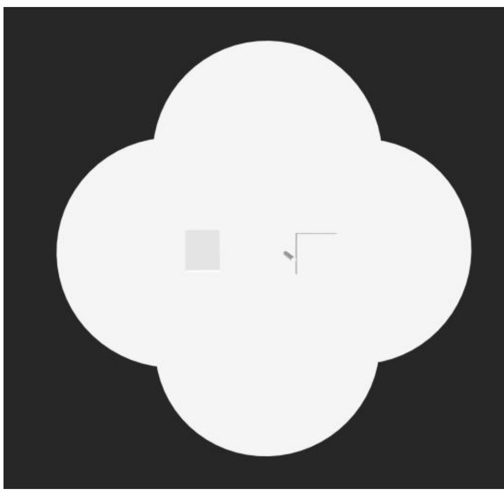
Figures

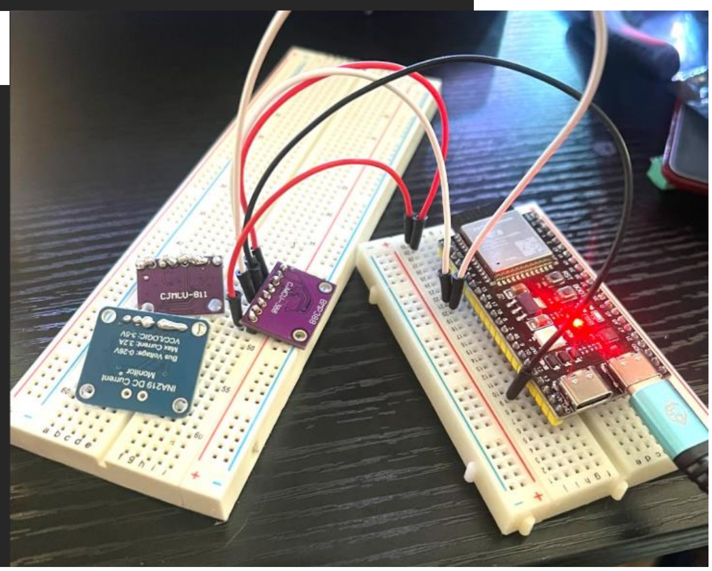

3.3 V Power On LED
USB-to-UART Bridge
ESP32-S3-WROOM-1/1U/2
RGB LED
Pin Headers
USB Port
Reset Button
Boot Button
USB-to-UART Port
5 V to 3.3 V LDO
Pin Headers

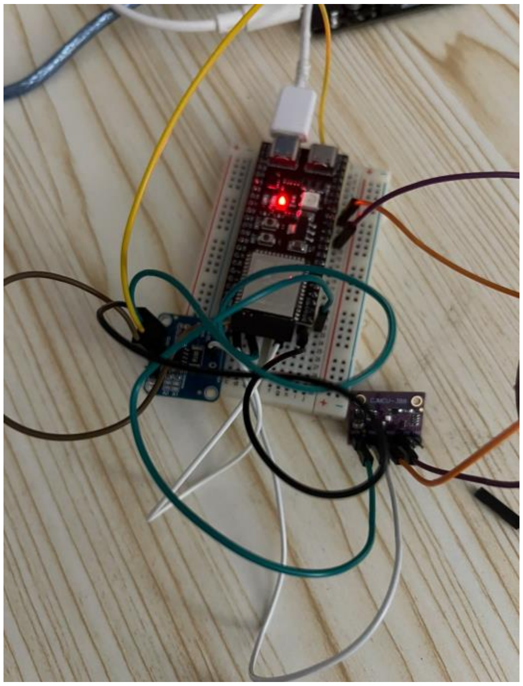

GPIO 23
GPIO 19
GPIO 18
GPIO 5
CMJU—388
RandomNerdTutorials.com
CP2102
EN
BOOT


File Edit Selection View Go Run
AEMD
EXPLORER
Welcome
README.md
main.py
main.py (Index)
gui.py
Tkinter-Designer > build > gui.py > show_power_data
AEMD
.venv
AEMD_DATADASH_REPO
.vscode
AEMD_DATA_DASH
main.py
README.md
AEMD_ONBOARD_COMPUTER
Tkinter-Designer
.github
build
assets
gui.py
docs
gui
tests
tkdesigner
.flake8
.gitignore
CODE_OF_CONDUCT.md
LEARN.md
LICENSE
Makefile
poetry.lock
pyproject.toml
README.md
requirements.txt
Screenshot 2025-04-08 214236.png
settings-icon-2048x2046-cw28eevx.png
OUTLINE
OUTPUT_PATH
relief="ridge"
)
canvas.place(x=0, y=0)
# Draw background base layer
canvas.create_rectangle(0.0, 0.0, 480.0, 320.0, fill="#0B0F1A", outline="")
# -- BUTTONS --
button_logs = Button(text="Logs", bg="#00FFD0", fg="#FFFFFF", font=("Rajdhani", 14), ...
button_live = Button(text="Live Feed", bg="#00FFD0", fg="#FFFFFF", font=("Rajdhani", 14), ...
button_data = Button(text="Data", bg="#00FFD0", fg="#FFFFFF", font=("Rajdhani", 14), ...
button_settings = Button(text="Settings", bg="#00FFD0", fg="#FFFFFF", font=("Rajdhani", 14), ...
buttons = [button_logs, button_live, button_data, button_settings]
def draw_main_screen(): ...
-- SELECTOR SCREEN FUNCTION --
def show_data_selector(): ...
def show_power_data(): ...
back_rect = canvas.create_rectangle(180, 270, 300, 300, fill="#FF5CAA", outline="")
back_text = canvas.create_text(240, 285, text="Back", fill="#FFFFFF", font=("Rajdhani", 14))
def back_to_selector(event=None): ...
canvas.tag_bind(back_rect, "<Button-1>", back_to_selector)
canvas.tag_bind(back_text, "<Button-1>", back_to_selector)
# -- RETURN TO MAIN SCREEN --
def return_to_main(): ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
python - build






CJMLU-811



Ethical Responsibility

The project adheres to ethical engineering practices by:

- **Minimizing planetary contamination** in accordance with NASA Planetary Protection Protocols, ensuring compliance with international space exploration standards.
- **Promoting responsible AI integration** through the use of lightweight, interpretable models that prioritize transparency and operational safety.
- **Safeguarding mission-critical data** through secure communication protocols and edge processing strategies that reduce data vulnerability in high-risk environments.
- **Prioritizing system sustainability**, with a focus on energy-efficient design and long-term reliability in resource-scarce conditions.