ISBN: 978-628-95207-4-3. ISSN: 2414-6390. Digital Object Identifier: https://dx.doi.org/10.18687/LACCEI2023.1.1.1514

Development and Implementation of a Wireless Telemetry System for a Mechanical Rover Prototype

Alina Santander Vinokurova, Mechatronics Engineering[1], Tatiana Jaimes, Mechatronics Engineering[1], and Almaz Abdrasulov, Mechatronics Engineering[1]
[1]Vaughn College of Aeronautics and Technology, NY, USA
alina.santandervinokurova@vaughn.edu, tatiana.jaimes@vaughn.edu, almaz.abdrasulov@vaughn.edu

Abstract*– This paper outlines the design, development, testing, and deployment of a telemetry system aboard Vaughn College's Human Exploration Rover Challenge (HERC) rover to collect information on the environment, rover, and drivers' status. Developing this telemetry system allows the team to acquire data from a unique rover design and analyze it, simulating what a real space mission would be like. Based on the competition and design requirements, this system comprises a temperature and humidity sensor, an OV2640 camera, a Long Range (LoRa) module for communication, a Global Positioning System (GPS) sensor, and an ESP32 microcontroller. The system's transmitter acquires all the desired data, while the receiver displays the information on the Arduino Internet of Things (IoT) cloud for analysis.

Keywords — Telemetry, HERC, rover, ESP32, OV2640, LoRa, Arduino IoT

I. Introduction

Aerospace is a broad and challenging industry that allows humans to build different spacecraft, send them to outer space, and retrieve measured data for various analyses. For this reason, telemetry techniques have been quickly developed in recent years to obtain remotely measured data efficiently.

Telemetry systems are an alternative method to transmit data automatically from a remote source to an information technology (IT) system in a different location for monitoring and analysis. This system can be relayed through radio, Global System for Mobiles (GSM), satellite, or cable, and operates through sensors at the remote source to measure physical or electrical data. Telemetry is widely applied to satellites since it connects the satellite itself and the facilities on the ground. Its purpose is to ensure the satellite performs correctly through the monitoring of the health and status of the satellite through the collection, processing, and transmission of data from the various spacecraft subsystems, the determination of the satellite's exact location through the reception, processing, and transmitting of ranging signals, and the proper control of satellite through the reception, processing, and implementation of commands transmitted from the ground [1].

The main benefit of implementing a telemetry system is that it allows a user to monitor the state or environment of an object while being physically far from it. Telemetry is a valuable tool for ongoing performance monitoring and management, particularly for providing real-time data for improvements on future designs.

Digital Object Identifier: (only for full papers, inserted by LACCEI).
ISSN, ISBN: (to be inserted by LACCEI).
DO NOT REMOVE

Now, this highly useful technique for data collection in the aerospace industry can be implemented for smaller spacecraft such as a rover; more specifically, a human-powered rover for NASA's HERC competition. This competition's primary objective is for student teams to design, develop, build, and test a human-powered rover and a task tool capable of traversing challenging terrain and completing mission tasks along the course's path, respectively. The competition requires two students to use the designed vehicle to traverse a course of approximately half-mile that includes a simulated field of asteroid debris, boulders, erosion ruts, crevasses, and an ancient streambed [2]. Each team earns points by successfully completing obstacles and tasks, and the team with the highest accumulated number of points throughout the project year will be the winner.

II. Need Statement

Due to the significant distance between the source, the measuring end, and the receiver, delays often lead to system failure, requiring critical changes to the telemetry system for improvement. Among these, it is found that 28 percent of the failures in telemetry systems are attributable to lead and electrode problems, 25 percent to battery depletion, 22 percent to mechanical or electronic component failures, 12 percent to inappropriate control settings and frequency mismatching, and 13 percent to miscellaneous difficulties [3]. The current project aims to design and implement a telemetry system on a unique human-powered rover, as outlined within the HERC requirements. Through this implementation, it will be possible to retrieve new real-time data and improve common failures. The telemetry system collects data from a unique mechanical rover to inform the team about the driver, environment, and vehicle status. This is achieved by implementing an OV2640 camera, a temperature and humidity sensor, and a GPS sensor to inform the base team about the respective status. Furthermore, the designed telemetry system and the rover developed by the Vaughn College Rover Club, newly established, and founded by the first author of this project, will join the NASA Rover Challenge Competition against other colleges for design ideas and problem-solving skills.

A. Societal Impact

Through the team's close involvement with the HERC competition with the telemetry system, the project's goal is presented in three parts. First, to provide a solid platform for Vaughn College to reach future generations of engineers by representing its college community in an international

challenge; second, to produce an environment for the new Rover Team members to expand their knowledge by exposing students to real engineering problems and applications through the implementation of a telemetry system and complete the challenge based on its requirements and constraints; and, lastly, to promote science, technology, engineering, and math (STEM) to over 200 middle and high school students through outreach events, required for the STEM engagement portion of the competition, to encourage young students to take part in engineering challenges. Furthermore, implementing the proposed telemetry system gives the students a clear example of data analysis and communication systems implemented in the aerospace industry, which is essential to track the status of missions in space.

This is a big competition for Latin American teams, as their participation has greatly increased since HERC's first race in the mid-90s. Through the telemetry system project and subsequent participation, the team expects to expand its community outreach to Latin American schools and institutions to provide training and workshops that offer sufficient foundations to start new teams in the continent for future competition participation.

B. Environmental Impact

The team aims to build a reusable and improvable telemetry system composed of assembled parts that are easy to change, replace, and repair. This ultimately extends the system's operational life and allows it to be reutilized and repurposed for future participation in the challenge.

III. Market Research

Although aerospace telemetry dates from the late 1930s, with the development of the balloon-borne radiosonde – measured meteorological data – that sent information to an Earth station by radio and is currently exemplified in observatory satellites that have performed as many as 50 different experiments and observations over the last decades; developing these telemetry systems are commonly led by large aerospace companies and, due to non-disclosure agreements (NDAs) for security purposes, little technical information is found online to recreate a simplified and inexpensive version of this system. Compared to the market, the team's telemetry system is a personalized and brand-new design targeted to aid in the data collection of a unique rover at the yearly HERC competition.

IV. Audience

The team's telemetry system is targeted at Vaughn College's NASA Rover Club student members to collect status information from their rover's driver, the environment, and the vehicle for training and competition purposes; NASA HERC judges who will assess for design awards, and the Latin American teams that will utilize the team's findings as a reference for future implementation at HERC competitions.

V. Engineering Requirements

The team's telemetry system must meet the following requirements to meet its data collection purpose according to team and competition standards:

1) *Continuous broadcasting:* The system's video and sensor data must be broadcasted and received by the central station to be accessible for real-time viewing.
2) *Stable video transmission:* The system's camera must transmit at least 12 frames per second (fps) to relay a stable video stream.
3) *Fast transmission:* The system's data must be updated to the cloud every one to three seconds to maintain accurate readings of the dashboard's display.
4) *Long-range transmission:* The network from the transmitter to the receiver must range from 300 to 400 meters, equivalent to the furthest point from the rover to the central station during the competition's course.
5) *Long-range communication:* The system's wireless communication module must have a minimum bandwidth of 433 MHz.
6) *Adaptable:* The system's temperature sensor must be capable of reading data ranging from approximately negative 30 °C to 60 °C.
7) *Compact design:* The transmitter circuit must fit within a housing of approximately 25 centimeters in length by 20 centimeters in width by 5 centimeters in height to allow a swift and secure mounting onto the rover.

VI. Budget

The total cost of the project, which entails the sensors and other components acquired for the telemetry system, are listed in Table 1.

Table 1
Component Price List

Component	Qty.	Subtotal USD
9V Battery	2	7.99
BME680 Sensor, Temperature Humidity Sensor	1	21.54
Raspberry Pi 3 Model B+	1	63
HiLetgo GY-NEO6MV2 Module	1	8.99
ESP-32S Board (3Pcs)	1	16.99
LoRa SX1278 RF Wireless	1	12.39
ESP32-CAM Camera Wi-Fi + Bluetooth Module	1	19.99
2Pcs OLED Display Module	1	9.99
40Pcs 2.54mm Breakaway PCB Board 40 Pin Header Connectors	1	12.95
Breadboard Jumper Wires	1	9.96
1kg PLA for housing prototype	1	22.99
Total		**206.78**

VII. Design Concept

The telemetry system is equipped with all the necessary electrical components to retrieve information from the environment, the vehicle, and the driver.

TABLE 2
DATA COLLECTION

Collection	Data
Environment status	Humidity, Temperature, Pressure
Rover Status	Location, Orientation, Speed
Driver Status	Visual information from driver's face

The user-friendly interface chosen to display the collected data allows to clearly identify the status of the simulated mission.

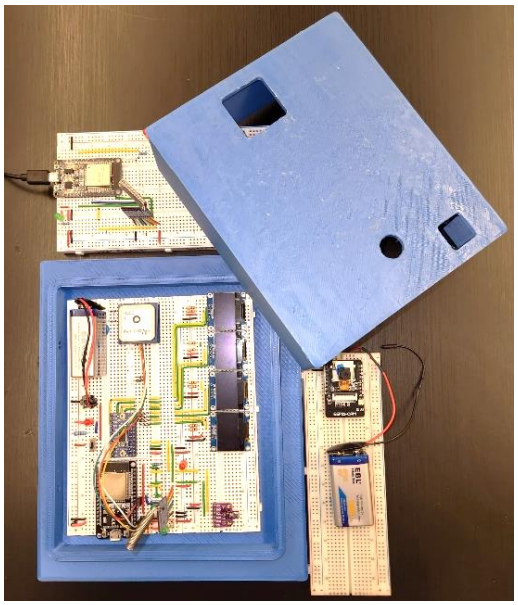

Fig. 1 Telemetry system

A. *User Interface*

Arduino's IoT cloud allows access to real-time data and display it on dashboards and gadgets, as seen in the figure below. This online software was chosen to show the data collected by the telemetry system, such as GPS location, speed, and temperature.

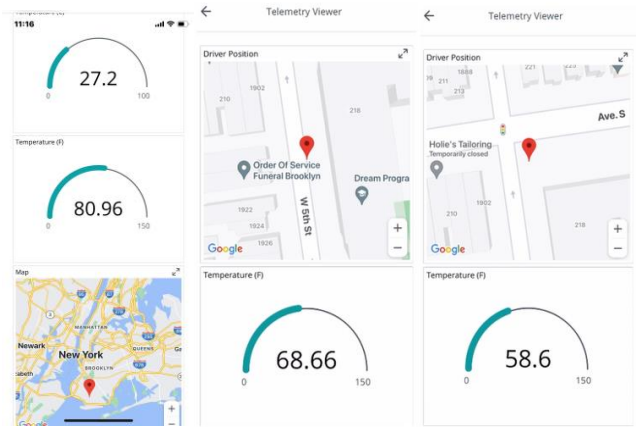

Fig. 2 Arduino IoT web display

B. *Electrical*

The electrical design consisted in a transmitter, a receiver, and a camera circuit.

1) *Transmitter*

This circuit collects all the information through the sensors and sends it to the receiver through the long range (LoRa) module. Fig. 3 shows the design of the transmitter.

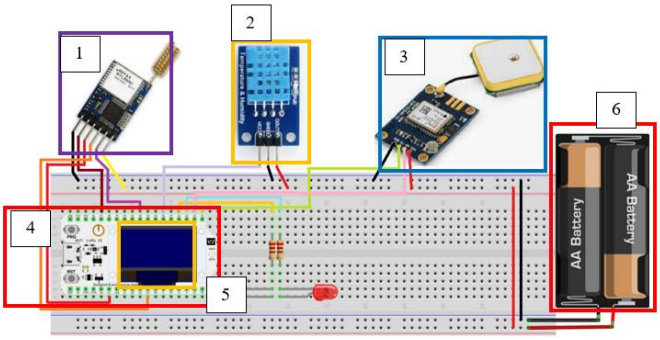

Fig. 3 Model of the transmitter's circuit design

Where:

1. LoRa for wireless communication
2. Temperature and humidity sensor (before choosing the BME680 sensor)
3. GPS module for testing and validation of positioning
4. ESP32 microcontroller, the "brains" of the operation, update from the Arduino due to higher memory capabilities.
5. OLED display to demonstrate data acquired.
6. Power supply of the system

All circuit components were connected, tested, and validated in different opportunities. Fig. 4 shows the built circuit for the transmitter.

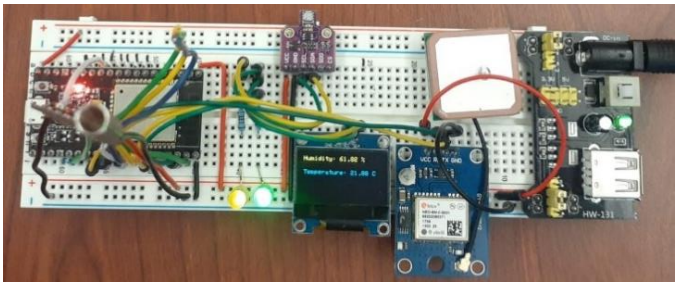

Fig. 4 Implementation of the transmitter

2) *Receiver*

As mentioned before, this circuit receives the information from the transmitter and connects to the Arduino IoT to display the information for the user. Fig. 5 shows the

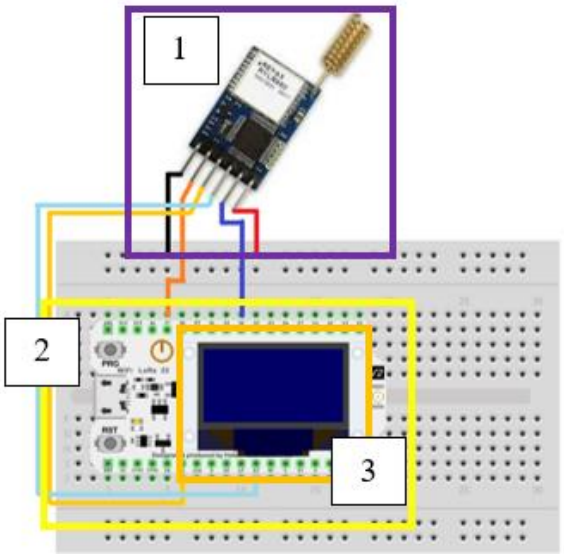

Fig. 5 Model of the receiver's circuit design

Where:

1. LoRa for wireless communication
2. ESP32 microcontroller
3. OLED display

The different components of the circuit were connected and tested to ensure its proper functioning. Fig. 6 depicts the final product of the assembled circuit.

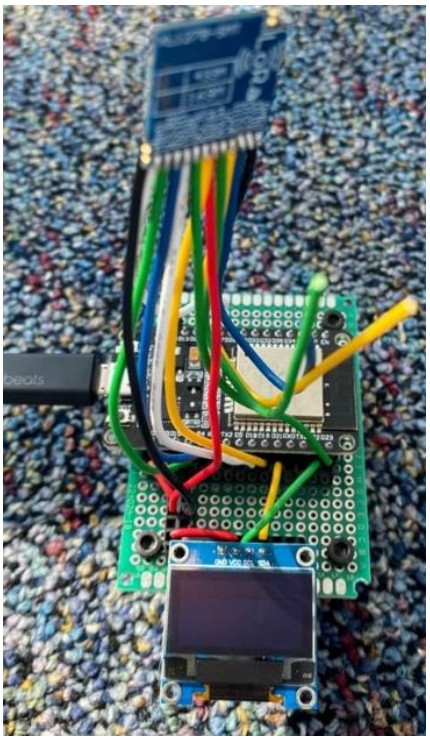

Fig. 6 Implementation of the receiver circuit

3) Camera

To monitor the status of the driver during the competition, an ESP32-CAM was implemented. The circuit is shown in Fig. 7. The video recorded from the camera was streamed online, and users with the link were able to access it.

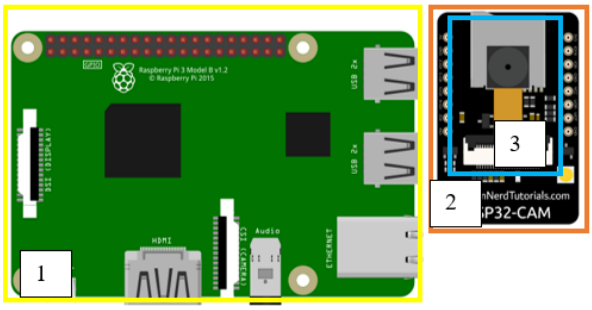

Fig. 7 model of the camera's circuit design

Where:

1. Raspberry Pi 3B+ that connects to the camera to provide power and run the ngrok agent to create the tunnel for global network access to the video stream.
2. ESP32-CAM, the microcontroller of the system
3. OV2640 that streams the video feed.

This circuit was tested and validated. Fig. 8 shows the implementation.

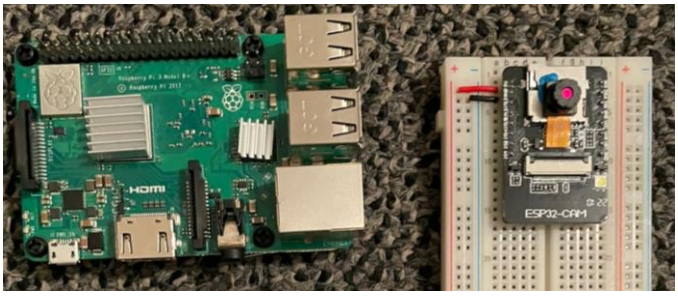

Fig. 8 Implementation of the camera circuit

C. Mechanical

The housing design encapsulates the current shape and size of the circuit on the breadboard. The cover has holes for the modules and sensors, such as the GPS, LoRa, and temperature and humidity sensors. This housing is 23 cm by 19 cm, and 5 cm tall. The base is extended so it can be attached to the frame of the rover, and both elements are attached by 4 screws.

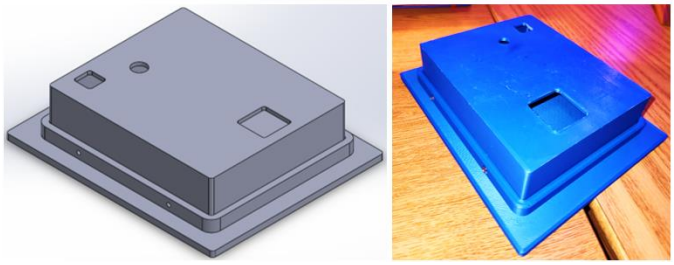

Fig. 9 housing design

D. *Coding*

1) Transmitter

The programming algorithm process of the transmitter unit is outlined below in Fig. 10, through which the system's microcontroller communicates with the peripheral devices using serial peripheral interface (SPI), inter-integrated circuit (I2C), and universal asynchronous receiver/transmitter (UART) protocols.

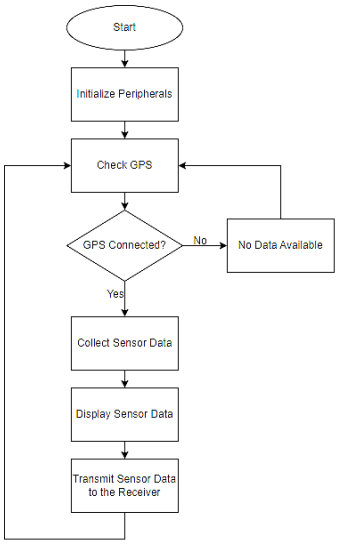

Fig. 10 Transmitter unit flowchart

After initializing the peripheral devices, the program checks the GPS sensor's status. If locked, the microcontroller collects the sensor data, displays it to the driver, and sends it to the receiving station via LoRa; otherwise, no data is available or recorded. Once the GPS is online and functioning as expected, the data collection portion of the system begins. And the necessary libraries – including SPI, LoRa, Adrafruit_BME680, TinyGPS, and HardwareSerial header libraries - were added to the program, as illustrated in Fig. 11, which enabled a successful connection to the sensors, displays, and modules.

```
#include "thingProperties.h"
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h> // I2C for BME680 and OLED
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <HardwareSerial.h>
#include <TinyGPS++.h>

//Pin definitions
#define CHIP_SEL  5
#define RST 14
#define DIO_0 2
#define LED 15

HardwareSerial neogps(1);
TinyGPSPlus gpsData;
Adafruit_BME680 bme;
```

Fig. 11 Libraries included in Transmitter Unit

Four OLED displays were added to the unit to provide sensor and position data to the driver, and the position and display settings, including the text color, size, cursor, and print options, were established through the Arduino IDE. For the proper execution of the transmitter unit, six void functions – TCA9548A, gps_update, displayHumid, displayTempF, displayTempC, and displayPressure – were created to collect all the information of the environment, the rover, and the drivers' throughout the course.

The *TCA9548A* void function is a one to eight I2C multiplexer that allows to expand the I2C bus, useful when controlling multiple I2C devices with the same I2C address. The *gps_update* function manages the GPS sensor's data collection to display the rover's speed in miles per hour, altitude in meters, latitude and longitude coordinates, and the number of satellites connected to the sensor to obtain the rover's location. And, finally, as the name suggests, the *displayHumid*, *displayTempF*, *displayTempC*, and *displayPressure* void functions oversee the data collection for the humidity in percentage, the temperature in Fahrenheit, the temperature in Celsius, and the pressure information, in kPa, of the environment, respectively.

Furthermore, these functions are implemented within the system's main setup program, illustrated in Fig. 12.

```
void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(115200);
  // This delay gives the chance to wait for a Serial Monitor
  delay(1500);

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();

  pinMode(LED,OUTPUT);
  neogps.begin(9600, SERIAL_8N1, 16, 17);
  bme.begin(0x77);
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  Wire.begin();

  while (!Serial);
  Serial.println("LoRa Sender");
  LoRa.setPins(CHIP_SEL, RST, DIO_0);
  delay(1000);
```

Fig. 12 Transmitter unit void setup

The void setup, displayed in Fig. 12, is the first function to be executed, only once in the sketch. All peripheral devices are initialized in this setup, and the I2C protocol, GPS module, environmental sensor, and LoRa transmitter are set up. Likewise, the Arduino Cloud connection is established to begin the data transmission from the transmitter unit to the central base computer. The OLED displays communicate with the system's microcontroller via I2C. And a multiplexer is used to allow each display to demonstrate different information since all the display components contain the same address.

2) Receiver

The programming algorithm process of the receiver unit is outlined below in Fig. 13, in which, after initializing the IoT properties, the microcontroller connects to the Internet via Wi-Fi. Once the Wi-Fi is connected, the program checks if LoRa is available and, if so, the program reads the received data and outputs it to the dashboard for the team to visualize it.

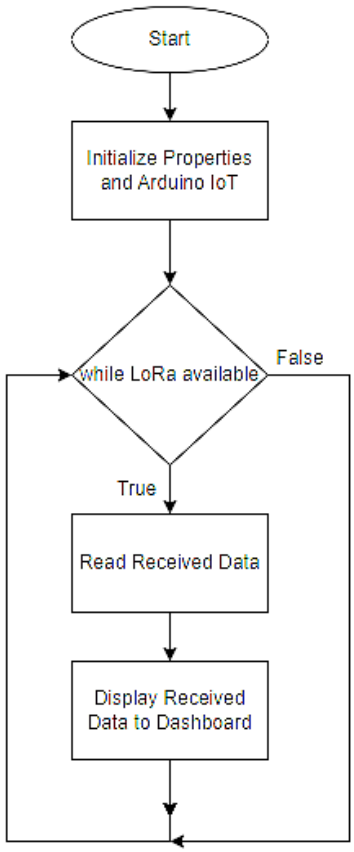

Fig. 13 Receiver unit flowchart

The Arduino IoT utilizes variable synchronization that allows devices to communicate with each other by sharing variables. This feature is called a cloud variable, which can link variables of the same data type between two or more devices. The variable is linked between a multipoint control unit (MCU) and the Arduino IoT cloud. If a variable is updated on the MCU board, the Arduino Cloud will also receive this value, and vice versa. With this, the Arduino IoT Cloud automatically ensure that any change to their value is propagated among all linked devices.

The void setup function for the receiver, displayed below in Fig. 14, includes an initialization of the Serial Monitor and properties using the Serial.begin(9600); and initProperties; commands. The Arduino IoT Cloud connection is also initialized using the built-in 'begin' function, which uses methods from the Arduino IoT Cloud and the Arduino_ConnectionHandler libraries, included in the thingProperties headerfile. Finally, setDebugMessageLevel; and ArduinoCloud.printDebugInfo; are used for debugging, as it prints information related to the network state and IoT Cloud connection and errors.

```
void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking
  delay(1500);

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
 */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
```

Fig. 14 Receiver unit void setup

Since the receiver unit's main purpose is to relay and confirm the information incoming from the transmitter, this portion of the system only utilizes one void function within the overall setup, loraRead, exemplified below in Fig. 15.

```
void loraRead()
{
  // try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize)
  {
    // received a paket
    Serial.println("");
    Serial.println(".....................................");
    Serial.println("Received packet: ");

    // read packet
    while (LoRa.available())
    {
      ArduinoCloud.update();
      digitalWrite(LED, HIGH);
      String humidity_raw = LoRa.readStringUntil('|');
      Serial.print("Humidity: ");Serial.println(humidity_raw);
      humidity = humidity_raw.toFloat();

      String tempF_raw = LoRa.readStringUntil('|');
      Serial.print("    Temperature: ");Serial.println(tempF_raw);
      tempF = tempF_raw.toFloat();


      String speed_raw = LoRa.readStringUntil('|');
      Serial.print("Speed: ");Serial.println(speed_raw);
      speed = speed_raw.toFloat();


      String satellites_number_raw = LoRa.readStringUntil('|');
      Serial.print("Number of Satellites: ");Serial.println(satellites_number_raw);
      number_of_satellites = satellites_number_raw.toInt();

      String altitude_raw = LoRa.readStringUntil('|');
      Serial.print("Altitude: ");Serial.println(altitude_raw);
      altitude = altitude_raw.toInt();

      String pressure_raw = LoRa.readStringUntil('|');
      Serial.print("Pressure atm: ");Serial.println(pressure_raw);
      pressureKPa = pressure_raw.toFloat();
      digitalWrite(LED,LOW);
```

Fig. 15 loraRead function setup

The loraRead function checks if the LoRa is available for the program to read the incoming string message constantly. The message is then converted to a floating-point type using the toFloat built-in function. From this, the Arduino IoT accesses the incoming string and uploads it to the Arduino IoT Cloud dashboard for visualization.

VIII. System Evaluation and Validation

All the telemetry subsystems were tested before their implementation, within the first three months of the project, to ensure the correct functionality of the components.

A. Sensors and components testing

1) Temperature and humidity sensor

The selected sensor to obtain the environmental status is the BME680 sensor. This is the next-generation environmental sensor manufactured by Bosch. This sensor has temperature, humidity, barometric pressure, and volatile organic compounds (VOC) sensing capabilities [4].

The sensor was connected to the microcontroller and OLED display, to obtain the environment's temperature and humidity information to then compare it with other sources. The sensor's accuracy was validated as seen in Fig. 16.

Fig. 16 Receiver unit's humidity and temperature readings

2) LoRa module

The module utilizes an SX1272 module, with a dual frequency band of 902-928 MHz, a transmission power of 25 mW, a sensitivity of -134 dBm, and a range of approximately 2 km [5]. LoRa's range was tested from various distances and heights to determine the sensor's constraints due to obstacles between the transmitter and receiver.

The highest range obtained, while the transmitter and receiver were at the same height, was 200 meters. Since this number was significantly shorter than the one expected, by at least 300 meters, the team decided to try different heights to increase the range.

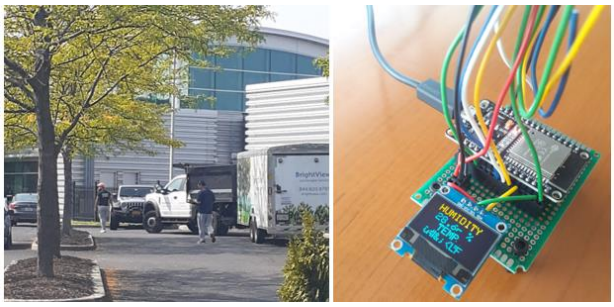

Fig. 17 Testing LoRa's range at the same altitude

The highest range obtained, while the transmitter and receiver were at different heights, was 600 meters, which increases the module's range than when the circuits were at the same altitude.

Since the longest range that the LoRa can successfully communicate between modules is 600 meters, and the distance from the furthest points of the competition course adds up to 300 meters, the communication module is well between the system's capability, as seen in Fig. 18.

Fig. 18 Distance from the furthest points of the course

3) GPS

When this sensor is activated and connected to the system's network, it displays the number of connected satellites to communicate said information. Since the longitude and latitude originally displayed its location accuracy to the second place, the code was updated so that the float information was accurate to the sixth place, six decimals after the flat point. This greatly improved the GPS accuracy, as observed in Fig. 19 and 20.

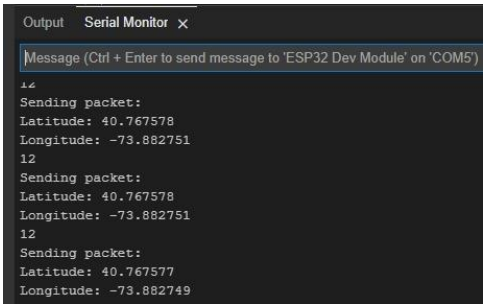

Fig. 19 Positioning information received from the transmitter

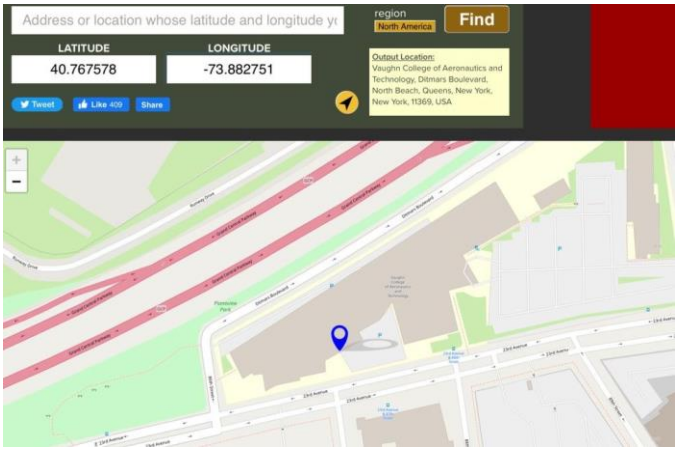

Fig. 20 Location coordinates where testing was conducted

B. *Software testing*

The environmental and rover data was collected and displayed on the Arduino cloud dashboard, through a Wi-Fi connection, regarding the temperature, humidity, pressure, speed, and location measured by the transmitter. This information was utilized to visualize the data from the team's computers and smartphones.

Resulting connection issues were solved through further testing by using mobile data and different networks.

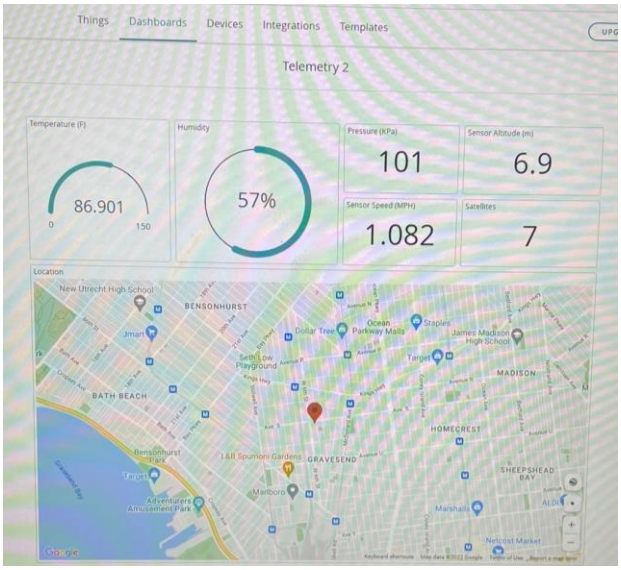

Fig. 21 Project application of the IoT cloud display

C. *Camera validation testing*

Only one device could view the video feed at a time, from the local network link created by default, as long as the device was connected to the same Wi-Fi or Hotspot as the camera, from the local network link created by default. However, at the time, that was not an issue since only one device was needed to monitor the driver's status through the camera.

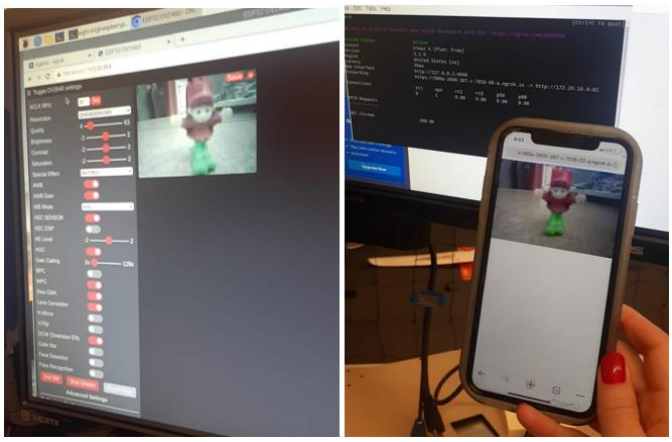

Fig. 22 Validation of code and URL for the camera

Nonetheless, the team went forward with troubleshooting the video feed's global access for multiple team members to monitor the drivers during the competition. A way to access the ESP32-CAM worldwide was using ngrok, a cross-platform application that enables developers to expose a local development server to the Internet easily. This software makes the locally hosted web server appear to be hosted on a subdomain of ngrok, meaning no public IP or domain name on the local machine is needed.

When ngrok was called into the code, it created a global network that allowed any device from other networks to connect and view. Nonetheless, the computer or device establishing the network to access the video stream must be connected to the same network as the ESP32CAM. As such, Raspberry Pi was implemented to act as a minicomputer located on the other side of the camera to provide the global network location.

First, the Wi-Fi connection was established to then find the ngrok. The ngrok creates a tunnel, referred to as a link, accessible to others worldwide. Every time a new link is created, a different tunnel is subsequently created as illustrated in Fig. 23.

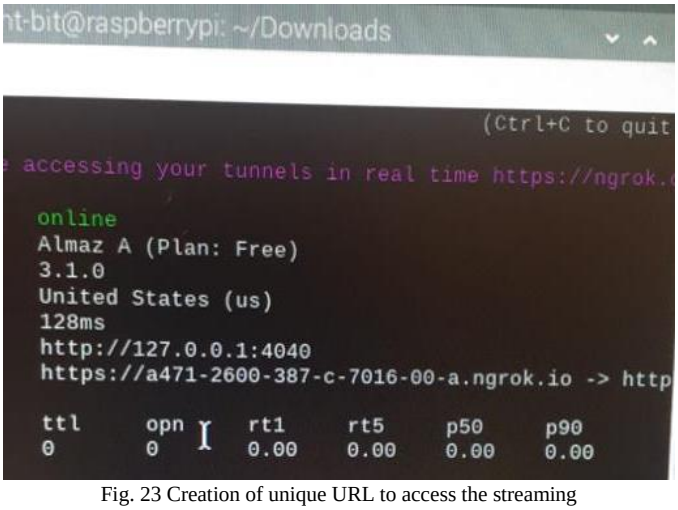

Fig. 23 Creation of unique URL to access the streaming

After ensuring a successful connection with the video streaming link, the system was validated by physically distancing the streaming device from the camera. The resulting video stream was stable, uninterrupted, and provided real-time information and status on the test subjects.

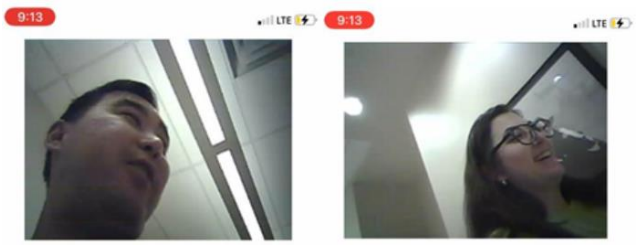

Fig. 24 Successful camera transmission testing

D. Housing design analysis

The telemetry system is placed under the back seat of the rover, as seen in Fig. 25, and mounted to the frame where two crew members will be placed throughout the front and the back of the rover during the competition.

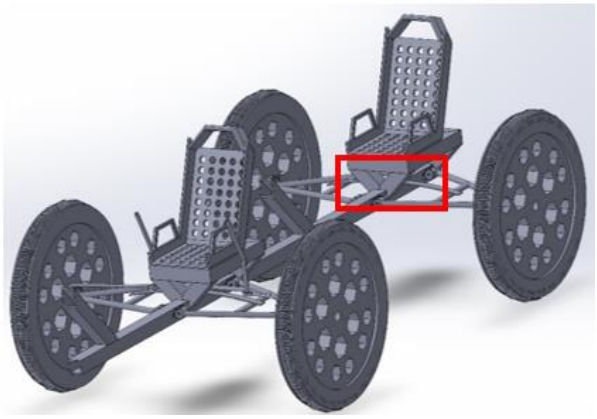
Fig. *25* Location of telemetry system on the rover

This causes the telemetry housing to undergo different forces to which the frame is exposed to. For this, the design was analysed on a CATIA simulation, where the load of the pilots was applied to the cover and the base separately, as seen in Fig. 26 and 27.

A force of 3000 N was applied to the base and cover, combining the pilots' approximate weight. Displacement and von Mises tests were performed to ensure the structure's integrity during the course; and the von-Mises stress was used to predict whether the telemetry system's housing material would yield or fracture.

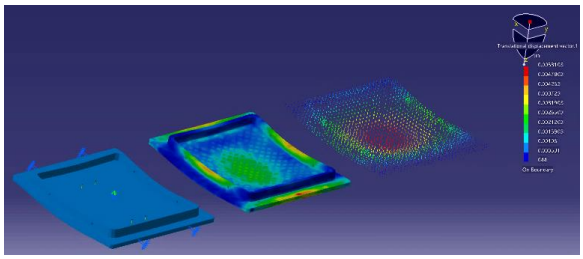
Fig. 26 Displacement test for housing base

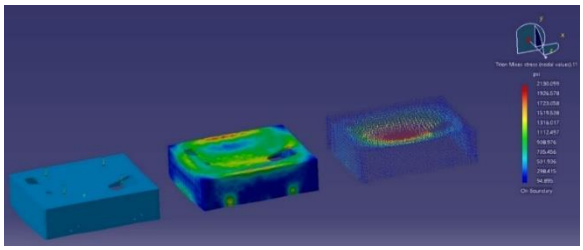
Fig. 27 von-Mises test for housing cover

The factor of safety, which is the ability of a system's structural capability to remain stable beyond its anticipated or actual loads, was calculated using the following equation:

$$Factor\ of\ Safety = \frac{Maximum\ stress}{Working\ or\ Design\ Stress} \quad (1)$$

The allowable stress of the 6061-T6 aluminium is 21,000 psi [6], and the system's housing design stress is 3,219 psi. Using this value, the factor of safety result is 6.5.

$$Factor\ of\ Safety = \frac{21{,}000\ psi}{3{,}219\ psi} = 6.5$$

A safety factor greater than one represents how much the stress is within the allowable limit. With a value of 6.5, the structure is validated to maintain its integrity.

IX. Conclusion

The telemetry system proved feasible to collect data for a mechanical rover for the NASA Rover Challenge, simulating a real mission in space and introducing college students to a new competition field.

The designing and testing phases of the telemetry system allowed the team to demonstrate and prove the project's practicality for real-world applications. Furthermore, this allowed the team to find future features that could be added to the project.

Among the future system improvements, the team intends to implement printed circuit boards (PCB) to make the system's structure lighter and easier to transport and utilize Artificial Intelligence (AI) to make decisions based on the collected data, useful to create competition strategies to maximize the team's time to complete the course.

ACKNOWLEDGMENT

The authors would like to thank the President of Vaughn College of Aeronautics and Technology, Dr. Sharon DeVivo, the Engineering and Technology Department Chair, Dr. Hossein Rahemi, and faculty professor Dr. Shouling He for their continued support throughout the whole project, and the Department of Education federal grant (Title III, Part F, HSI-STEM, and Articulation grant) which provided necessary funding support to engage Vaughn's students in STEM-related scholarly and professional activities. Also, a special thank you to Professor Donald Jimmo for his continuous support in putting this project together.

References

[1] Guest, A. N. (2013). "Telemetry, Tracking, and Command (TT&C)." In: Pelton, J.N., Madry, S., Camacho-Lara, S. (eds) Handbook of Satellite Applications. Springer, New York, NY. [Online]. https://doi.org/10.1007/978-1-4419-7671-0_69. [Accessed November 1, 2022].

[2] "Human Exploration Rover Challenge: About the Challenge." *NASA.* [Online]. Available: https://www.nasa.gov/stem/roverchallenge/competition/index.html. [Accessed March 15, 2023].

[3] Haslam, K. R. "Problems with telemetry monitoring systems." *PubMed.gov.* [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/10239827/. [Accessed February 20, 2022].

[4] "BME680." *Bosch, Invented for life*. [Online]. Available: https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme680/. [Accessed August 21, 2022].

[5] Felch, R. "Introducing LoRa (Long Range) Wireless Technology – Part 1." *Black Hills.* [Online]. Available: https://www.blackhillsinfosec.com/introducing-lora-long-range-wireless-technology-part-1/. [Accessed October 4, 2022].

[6] "Mechanical Properties of Various Alloys." *Knowledgebase - FAQ - Help Center - The Wagner Companies*. June 4 2018. [Online]. Available: https://kb.wagnercompanies.com/knowledge-base/mechanical-properties-of-various-alloys/. [Accessed January 2, 2023].