VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

SKY DADDY

SKY DADDY

PROJECT REPORT – NANOSATELLITE DESIGN AND ELECTRONICS

PROJEKT REPORT– DIZAJN A ELEKTRONIKA NANOSATELITOV

AUTHOR **Bc. MICHAL GLOS, Bc. JIŘÍ JÍLEK, Bc. TOMÁŠ HÁJEK Bc. DAVID HAMRAN**

Autor Bc. Michal Glos, Bc. Jiří Jílek, Bc. Tomáš Hájek Bc. David Hamran

BRNO 2024

1 Introduction

This project report introduces our initiative to address the challenge of gathering telemetry data using a flying system. We have developed a solution that involves programming receiver and transmitter components, integrated with sensors. To ensure the functionality and protection of our system, we have constructed a custom enclosure using 3D printing technology. This report provides an overview of our approach to solving this problem and outlines our progress in implementing and eventually testing the solution.

2 Proposal

2.1 Components

2.1.1 Telemetry

MPU-6500 (Gyroscope and accelerometer)

The MPU6500 sensor was chosen because it was already available in the lab. This Inertia Measurement Unit (IMU) provides data about the acceleration and orientation in 3 axes. The temperature can be read out as well, but its inaccuracy is prominent. The data from the IMU were used to determine the aircraft's dynamics and estimated position.

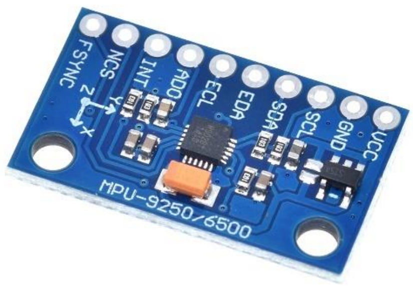


Figure 1: MPU6500 sensor.

BME280 (Humidity sensor, barometer and Thermometer)

To broaden telemetry data, we opted for BME280. Serving as a humidity, barometric pressure, and temperature sensor, it offers accurate measurements (accuracy of ±3 % for humidity, ±1 hPa for pressure, and ±1° for temperature) which supplements IMU data. Thanks to the I2C bus, the sensors were connected just to two pins of our MCU.

Figure 2: BME280 sensor.

2.1.2 On-board computer and Ground station

ESP32c3 (on-board computer)

The ESP32C3 was selected as an onboard computer for our project due to its compact design, power management circuitry, and communication capabilities, notably utilizing ESP-NOW technology. The main purpose of this OBC is to process data and transmit them in real-time to the ground station.

Its compact form factor and low power consumption make it ideal for integration into our payload, ensuring minimal impact on overall system weight and power requirements. Additionally, as it is a microcontroller made by Espressif Systems, it offers the ESP-NOW communication protocol. This protocol offers very lightweight and fast point-to-point communication with long-range capabilities, despite its operating frequency of 2.4 GHz and limited power output. The microcontroller was chosen based on previous experience with it.

ESP32 wroom32 (ground station)

We used the ESP32 development kit as our ground station. Since its only job was to relay data from the wireless ESP-NOW protocol to Serial we did not consider any other solutions or microcontrollers that we did not previously own.

2.2 Budget

Item	Cost	Bought
MPU6500	92 CZK	NO
BME280	98 CZK	YES
ESP32c3	298 CZK	YES
ESP32 dev kit	230 CZK	NO
Total	396 CZK	

2.3 Risk assessment

Risk	Importance Ranking
Equipment failure (sensor, communication, battery)	High
Adverse weather conditions during launch	High
Non-compliance with airspace regulations	Medium
Safety concerns during balloon launches	Medium
Data loss or corruption	Medium
Budget and resource constraints	Low

3 Implementation

3.1 Hardware configuration

All hardware was wired together on a protoboard PCB using wires. The MCU and IMU were placed into corresponding connectors, ensuring that they could be removed hassle-free and used for future projects. Because of size constraints, the environmental sensor was hard-wired in place. After we decided to use the plane instead of the balloon, we located the regulated 5 volts on the plane's transceiver and used it as a power source for our device.

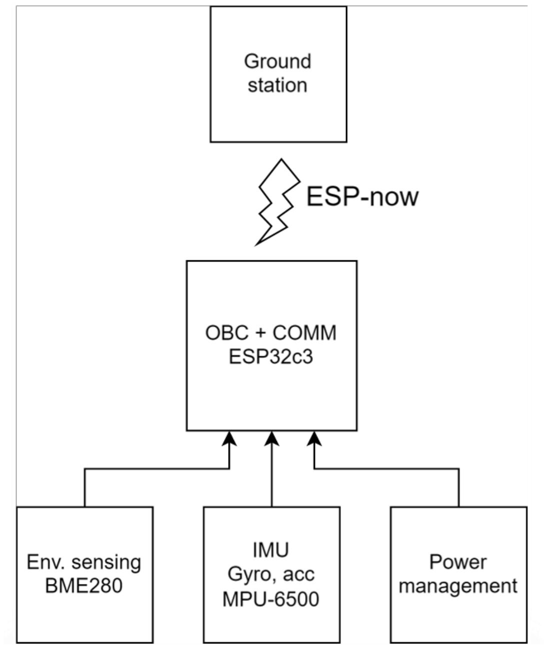


Figure 3: Hardware diagram.

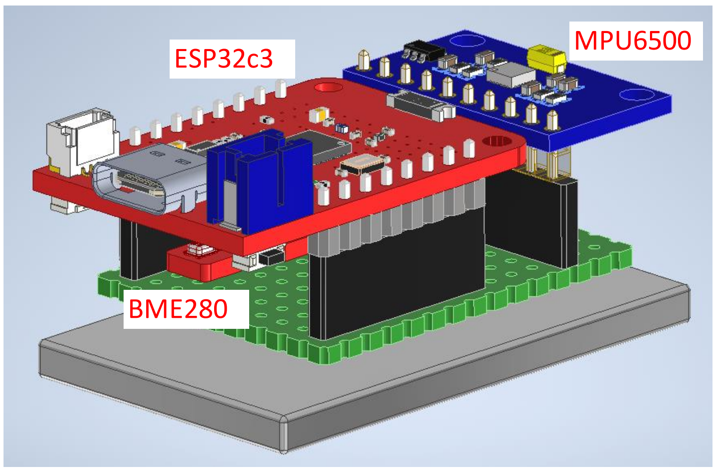


Figure 4: Whole assembly of the hardware in the CAD software.

3.2 Assembly

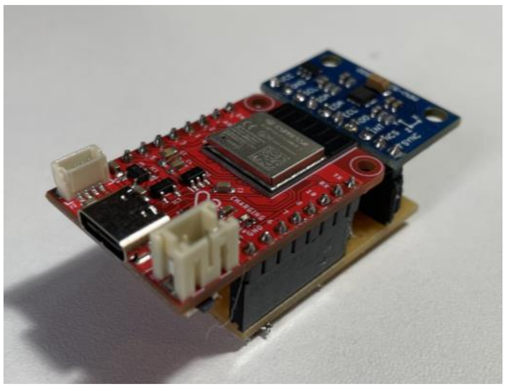


Figure 5: Completed hardware, top side view.

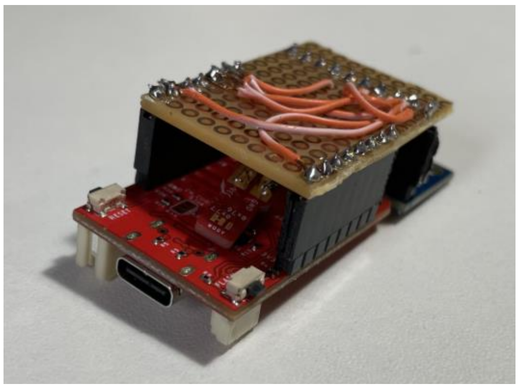
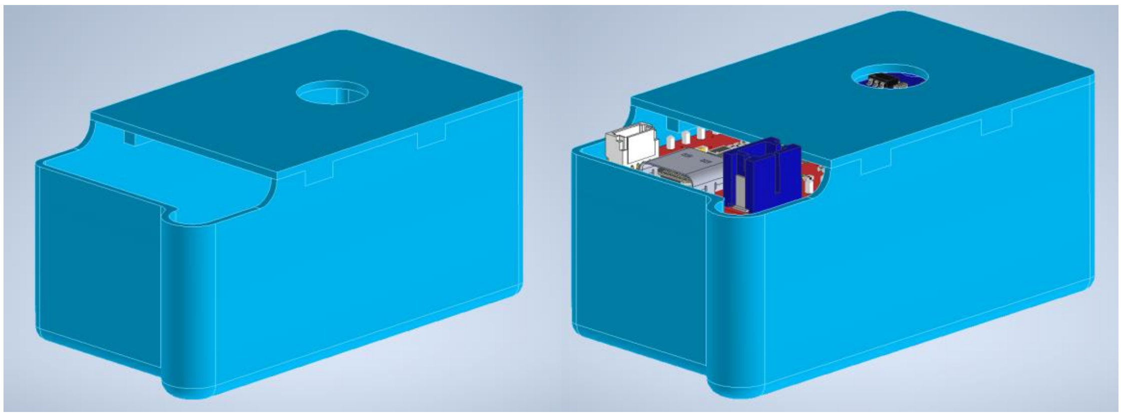


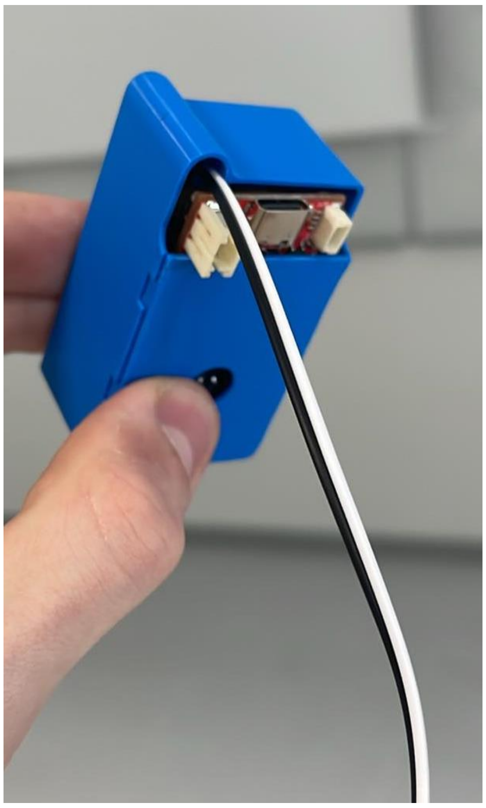
Figure 6: Completed hardware, bottom side view.

3.3 Box design

To achieve satisfying solution to our project, the design of the protective shell underwent several iterations. The blue box was initially created using AUTOCAD Inventor 2024 software by AUTODESK. The assembled hardware model was then implemented into the virtual model of the protective box to ensure that the designed models are dimensionally compatible.



The protective box model was then printed on an Original Prusa MINI+ 3D printer, using classic PLA filament as the material. This eco-friendly choice is biodegradable, made from corn or potato starch, or even sugarcane. Since our project considered potential impacts, the box was further padded with polymer foam to enhance its protective factor and ensure the hardware's safety. Such polymer paddings do enhance the system resistance to vibration and mitigate sensor noise introduction into the system.

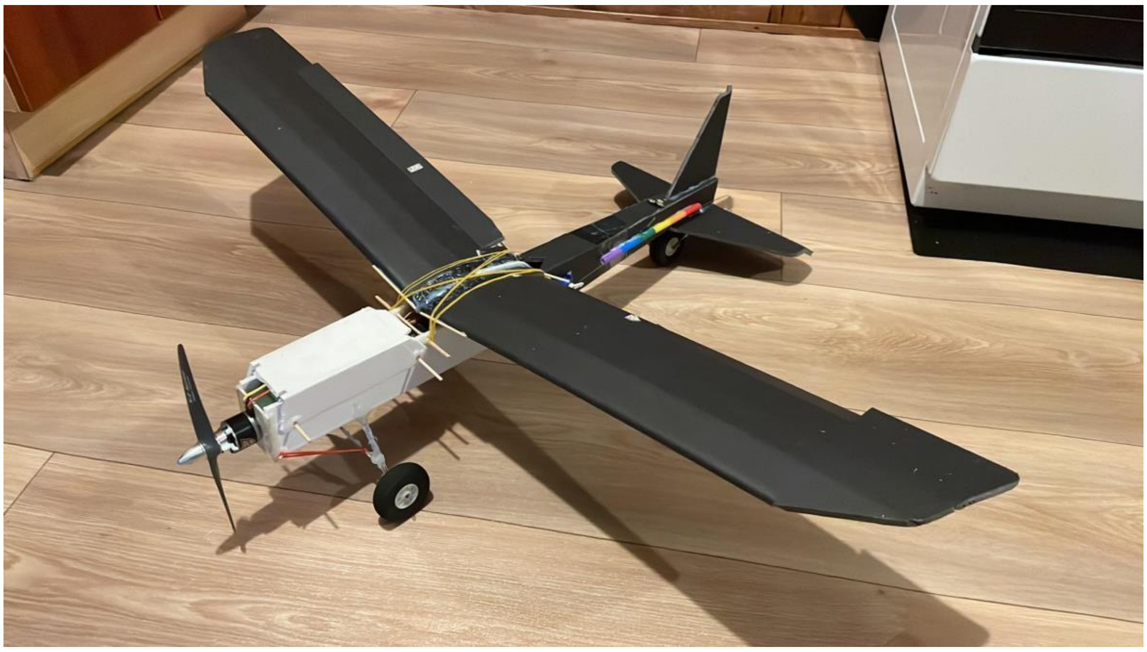


3.4 Mass budget

Components		Mass [g]	Total mass [g]
Gyroscope + accelerometer		2	**26.2**
Microcontroller		8.5	
Sensor		0.5	
PCB		3	
Structure	Box	7.7	
	Cover	2.5	

3.5 Vehicle of choice

For our vehicle, we chose an available RC plane. Our transmitter was adapted to get power from the Li-Pol 4S battery, already supplied with the plane. This choice was made on the premise – the helium for a balloon would be too expensive and hydrogen gas would be too dangerous.



3.6 Transmitter

The code for transmitter configures an ESP32 microcontroller to manage telemetry data transmission and reception. It initializes necessary libraries and sets up ESP-NOW communication.

Key global variables are defined, including broadcast addresses and message counters. Communication functions handle sending and receiving telemetry messages, while a helper function saves and sends messages from a buffer.

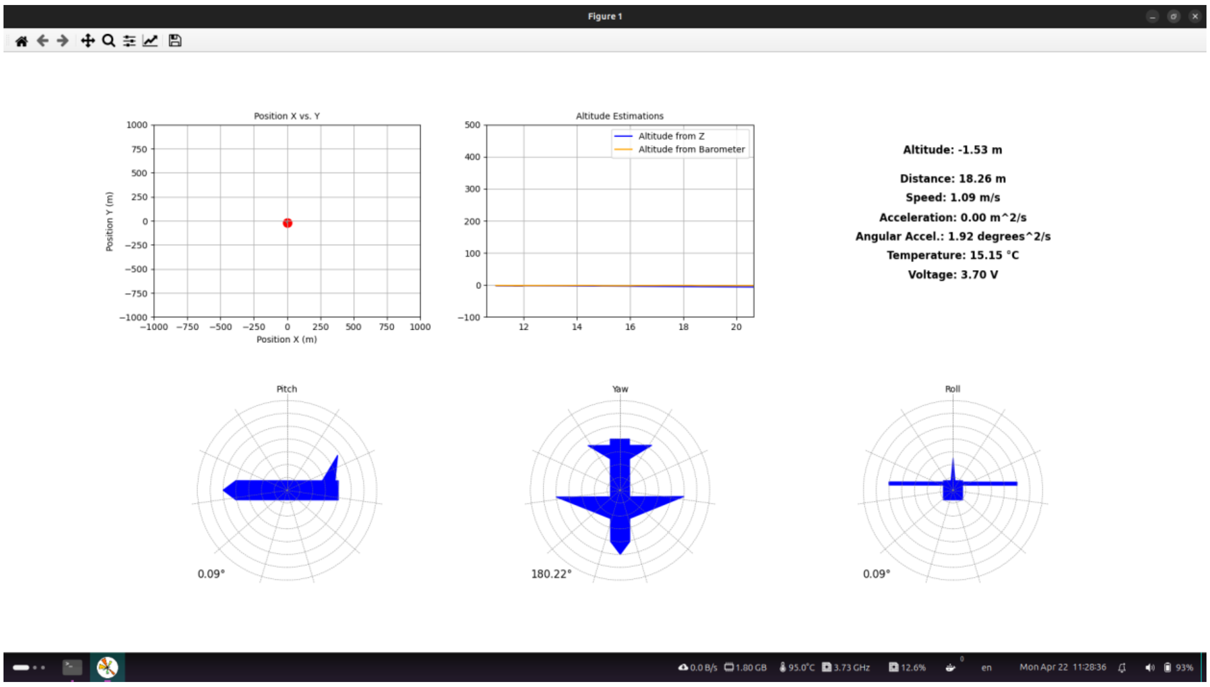
Additionally, sensor reading functions collect telemetry data, and periodic tasks manage communication and sensor processes. Setup functions initialize serial communication, WiFi, and ESP-NOW, while also configuring sensors and creating periodic tasks. Overall, the code orchestrates efficient telemetry data handling and communication within the system.

The transmitter itself holds the state of the system in terms of current orientation, position and particular derivatives obtained from the sensors. To fuse the gyroscope and accelerometer into orientation, we used the Madgwick algorithm (Arduino implementation). The quaternion rotation was applied to the accelerometer data to obtain the global frame of reference sensor data. With further integration, other values were calculated. Dynamic bias estimation was implemented for raw sensor reading to correct for drift.

3.7 Receiver

The code for receiver configures an ESP32 for receiving telemetry data. It initializes necessary libraries and sets up ESP-NOW for communication. Callback functions handle data write out to serial output and sending ack to the transmitter. When reset – the receiver starts the serial communication with CSV header, each other data packet received is considered a CSV data entry.

The receiver will be connected to a laptop via USB with a plotting frontend. Frontend provides visualization for flight and for testing – either raw data or a telemetry dashboard, shown below. The laptop will also generate a CSV file with the flight data after the reset button is pushed.



Altitude: -1.53 m

Distance: 18.26 m

Speed: 1.09 m/s

Acceleration: 0.00 m^2/s

Angular Accel.: 1.92 degrees^2/s

Temperature: 15.15 °C

Voltage: 3.70 V

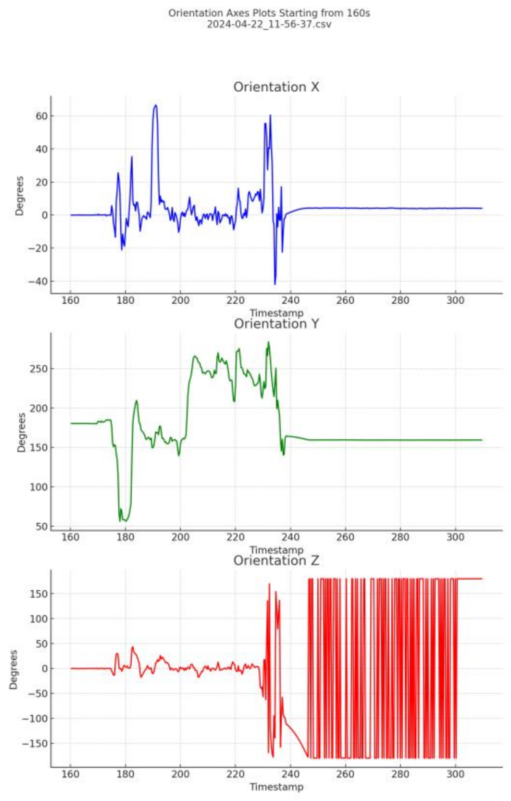
4 Outcomes

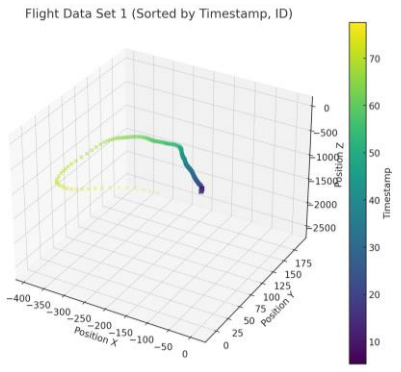
Although the issue of sensor drift was iteratively improved, the inherent instability of the inertial positioning system could not be mitigated. While the orientation vector calculation was optimized with Madgwick algorithm parameters and sensor bias correction, the position collected certain drift, mainly due to a little delay in the orientation vector update.

In total, 4 flights were conducted, collecting 4 CSV data files.

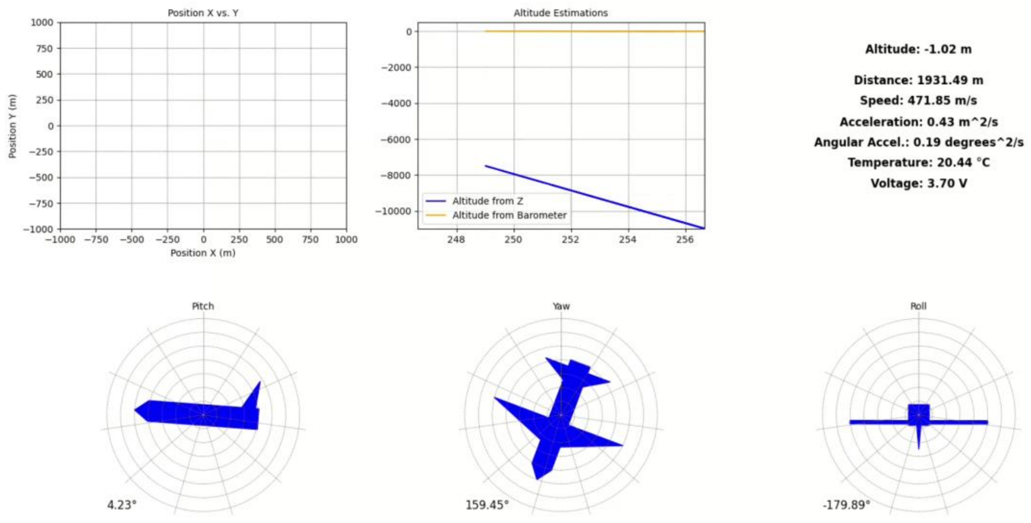
4.1 Measurements

Here, we can observe the position and orientation of the RC aircraft during one of such flights. On the orientation plot, we might observe the aurcraft was in an instable orientation position – 180˚ crashed on it’s top.





On this screenshot, we might observe the crash of the aircraft. The roll value is around 180° and we might also see the drift in the values of integrated sensor positions in the global frame of reference.



Altitude: -1.02 m

Distance: 1931.49 m
Speed: 471.85 m/s
Acceleration: 0.43 m^2/s
Angular Accel.: 0.19 degrees^2/s
Temperature: 20.44 °C
Voltage: 3.70 V

4.2 Lessons Learned

Local memory storage

Because of the inherent drift of the IMU, local memory with RAW data from the sensors would come handy. The iterations performed could have been simulated on the RAW data to obtain more accurate and precise state estimation. Also, in case the airicraft would get too far for too long, there would be no data loss.

GPS

Due to inherent drift in the inertial unit sensor integration, some global reference sensors, such as GPS or magnetometer, would be usefull to perform periodic correction of the system state to ensure accuracy of the system and mitigate the inertial drift.

Higher update frequency of the dashboard

Because only a plotting library for Python3 was used, the refresh rate for the telemetry dashboard was around 3-4 Hz, which is not satifactory. Using some dedicated software for desktop applications would certainly result in much better results.

5 Conclusion

This project effectively demonstrated a telemetry system for a remotely controlled aircraft, integrating economical yet functional sensors such as the MPU-6500 and BME280 with ESP32 microcontrollers. While challenges like sensor drift were noted, applying the Madgwick algorithm helped refine orientation data. The project showcased a reliable telemetry data stream that provided useful insights into the aircraft's flight dynamics and environmental conditions. Overall, this prototype successfully served its purpose, illustrating the feasibility of low-cost telemetry systems in RC aviation.