Critical Design Review (CDR)
CanSat Hungary 2025

Team: TFS

School Name & City: Baar-Madas/Trefort High School, Budapest
Date: 2025.02.17.

February 17, 2025

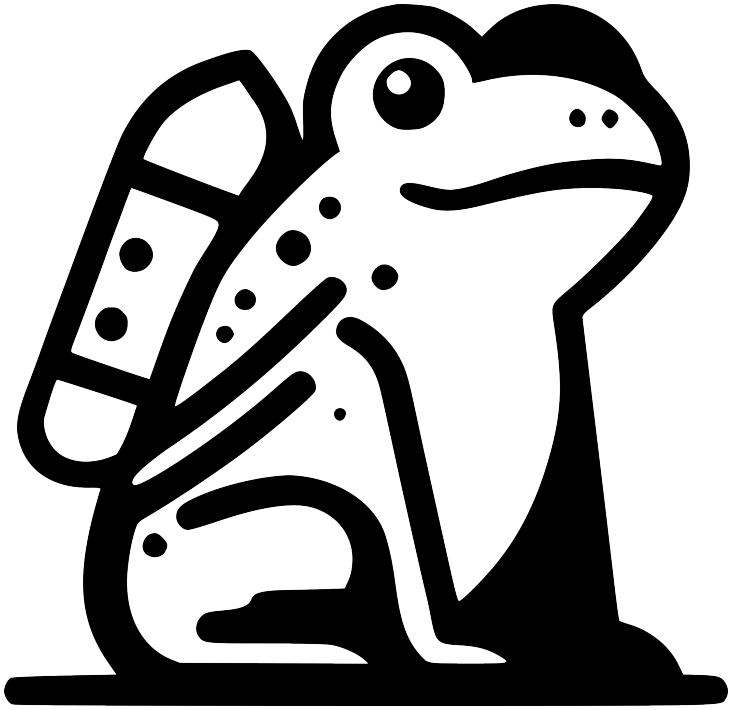


Table of Contents

1 Introduction

1.1 Introduction of the team

- **Csongor Vincze:** School: Baár-Madas Reformed High School. Tasks: team leader, organizing meetings, administrative tasks, mechanical design. Time: 250h
- **Luca Kern:** School: Baár-Madas Reformed High School. Tasks: implementation and testing of the parachute, managing group dynamics. Time: 150h
- **Attila Vincze:** School: Baár-Madas Reformed High School. Tasks: software development, researching/sourcing components. Time: 80h
- **Alfréd Burger:** School: Trefort Ágoston Practicing High School. Tasks: hardware design, construction, and mechanical implementation, software development. Time: 50h
- **Illés Fleischman:** School: Baár-Madas Reformed High School. Tasks: software development, social media, administrative tasks. Time: 220h
- **Benedek Bencz:** School: Baár-Madas Reformed High School. Tasks: development of mathematical/physical background, sourcing components, electrical design. Time: 200h
- **Károly Piláth:** Teacher at Trefort Ágoston Practicing High School. Mentor: assistance with professional questions, GNSS.
- **Csanád Budai:** Student at the Technical University of Budapest. Mentor: assistance with professional questions, control theory.

We must mention that our team consists mostly of 12th grade students so we think that it's only fair to face most of the challanges without the mentors. This explains the high working hours, we put in the project.

1.2 Mission objectives

On October 13th 2024 the fifth Starship flight test of SpaceX took place and successfully demonstrated the return and reuse system of the super heavy booster. Around this time we were brainstorming ideas for our secondary mission. We played with the thought of a targeted landing before, but when we saw the SpaceX test, we knew the real potential of this technology.

The primary mission of our CanSat is to measure temperature and pressure while descending with a parachute after being shot up with a rocket to around 1 km. It will also be sending these measurements down to the ground station via radio communication – as it is prescribed in the rules of the CanSat competition. Furthermore we measure humidity as well.

Our secondary mission is a targeted landing, using FPV drone motors and propellers. We control these motors with our GNSS based navigation software. With the help of simulations and measurements we hope to be able to predict how close the CanSat will get to the assigned point. By this, we mean that we will define a radius around the point within which the CanSat can land with a certain probability. With this we want to model how a real return system might work within the scope of the CanSat competition. As a bonus it will also help us find the Sat after landing.

2 CanSat Description

2.1 Overview of the mission

Our CanSat will be launched to around $1km$ by a rocket and will descend at around $5,5m/s$ with a parachute. During descent it will gather temperature and pressure data via a BME280 sensor and it will send this data down to the ground station with a WLR089 LoRa module. This is the primary mission of the CanSat as prescribed in the competition rules. The secondary mission is going to be a targeted landing achieved with BETAFPVs high performance drone motors. The onboard Raspberry Pi Pico is going to calculate the direction it needs to go in and control the motors, based on GNSS data from a GNSS 14 click module and an IMU unit with a Mahony AHRS algorithm to determine the yaw.

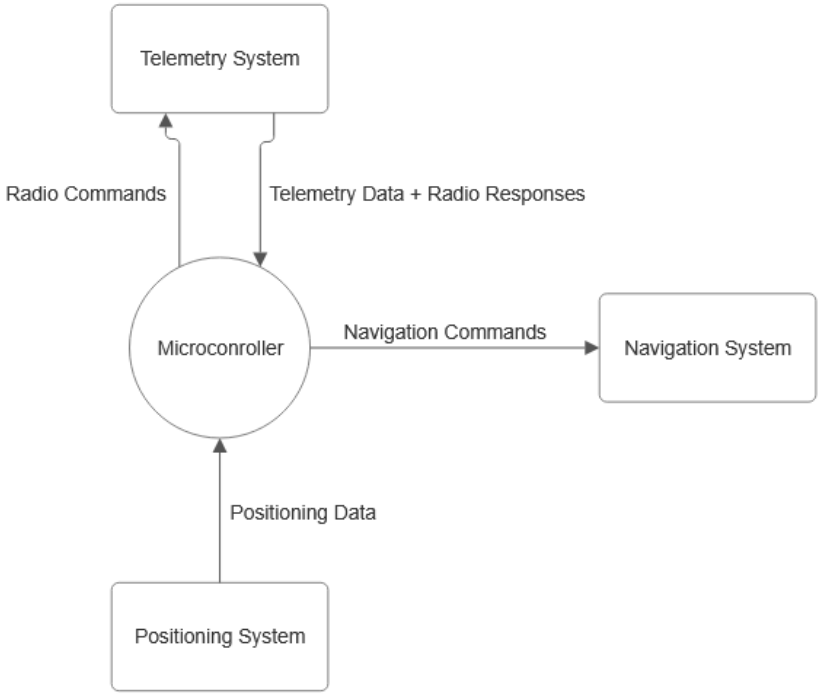


Figure 1: Block diagram of the different Systems

The design of the CanSat can be broken down into 3 essential systems.

- **The Telemetry System:** This consists of a BME280 temperature, pressure and humidity sensor and a WLR089 LoRa module. This is the system responsible for carrying out the primary mission of sending basic telemetry information back to the ground station.
- **The Positioning System:** This includes the GNSS 14 click GNSS module and the magnetometer to determine the cansat's location and orientation. This information will allow the CanSat to navigate itself which is essential for the secondary mission.
- **The Navigation System:** The CanSat has a parachute and 3 FPV drone propellers on the sides (not outside the 66mm diameter) angled 120 degrees from each other to be able to give horizontal speed to the CanSat in any direction while coming down. This combined with the information from the Positioning System will allow the CanSat to land as close to the target point as possible.

These systems are controlled by and communicate with each other via the on-board raspberry pi microcontroller and our micropython software.

2.2 Mechanical/structural design

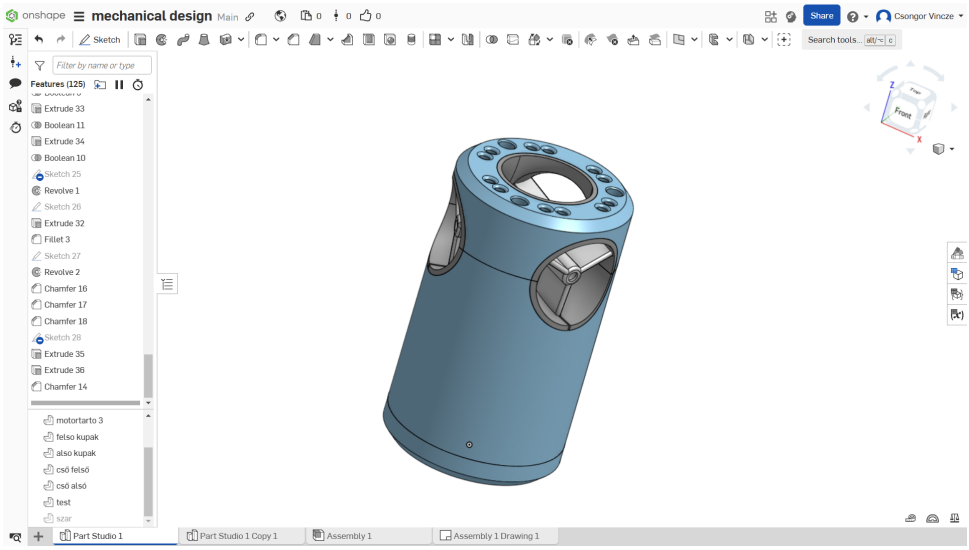
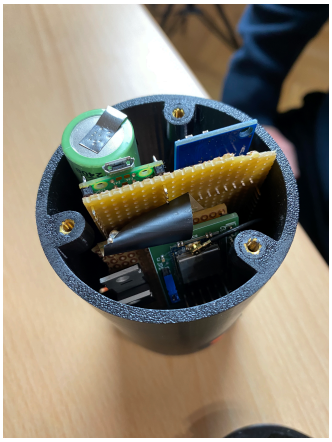


Figure 2: A picture of the full 3D design

The mechanical design of our CanSat consists of two main parts. The lower section houses most electronic components, while the upper section contains the motors. We will use custom PCBs to optimize space in the lower part; these will be screwed to the lower cap and will slide into rails in the housing. We have done testing to verify that we have enough space for all the components.

(a) The components



(b) The final configuration

Figure 3: Testing the fit of the circuit

It is possible to cut a hole for the antenna, thus we may plan a design with it sticking out of the body. Right now we are using the folding antenna, note that our LoRa tests were done with this one.

We initially 3D printed the housing with PLA, but parachute tests revealed that PLA may not withstand impact. We are exploring alternatives such as switching to PETG (due it's impact and heat resistance), adding foam to the bottom of the

CanSat to absorb energy, or even altering the housing geometry, but it seems like the current geometry with seelings and the foamy layer at the bottom will work properly.

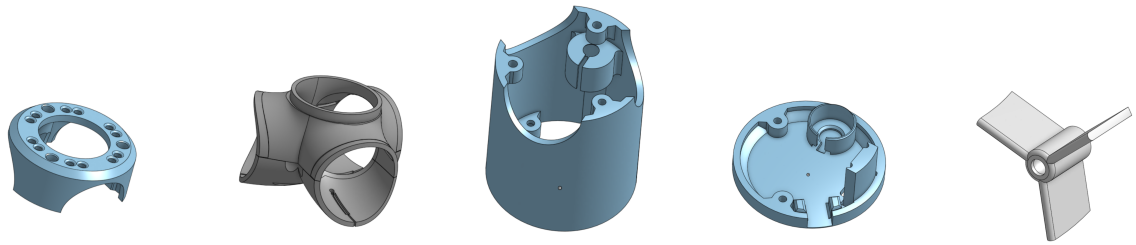


Figure 4: The 3D printed components of our CanSat

The housing consists of 7 3D printed parts: the lower cap, main body, tube, 3 motor mounts, and upper cap. The lower cap and main body include features to secure the battery. Since the battery is our heaviest component and is off center it might introduce an imbalance that we have to look into. Due to the fact that the mass of our CanSat is less than the required $300g$, the solution to the imbalance problem and the mass requirement will be the same. We are planning to put little masses to the other side of the housing of the CanSat. The lower cap also has openings for the charging port and power switch. The main body houses the electronics, while the motors are mounted inside the tube. The motor mounts are designed to hold the motors without glue, but they will be glued for extra security. All parts are fastened with screws except for the motor mounts, which are glued to the tube. We created the threads with heat inserts. Finally, the upper cap has holes – which are pairwise closer together – for attaching the parachute ropes.

We made measures to determine the weight and dimensions of the cansat. As an outcome of this test, we could conclude that we fit in the standards. In weight we have an extra $100g$ to fill, if we want to.

2.3 Electrical design

2.3.1 Components

In this section we will present the full schematic of our CanSat, and describe the operation of the components in detail from an electrical design perspective. Where measurements and calibration is required, we include the documentation for that as well.

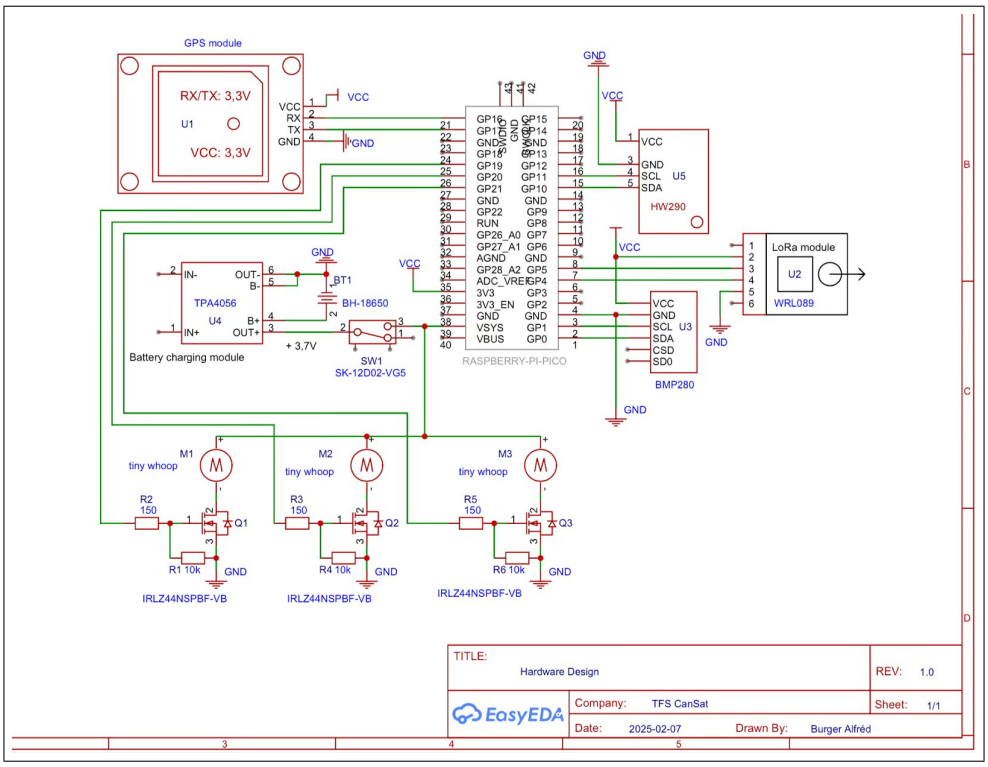


Figure 5: Circuit diagram of the electrical design

Microcontroller: The onboard hardware's central piece is a Raspberry Pi Pico microcontroller, which gathers data from our sensors and the GNSS module, and operates the radio and controls the engines.

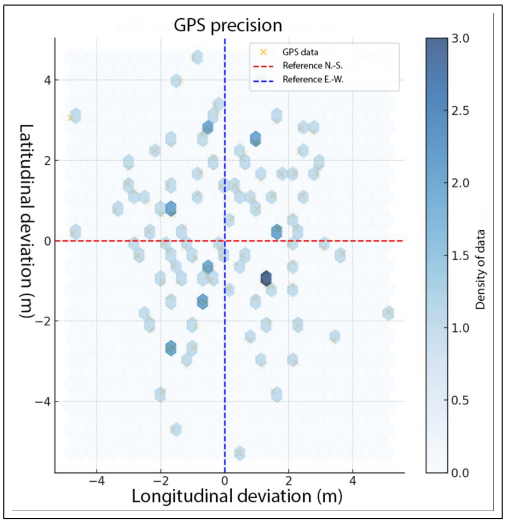
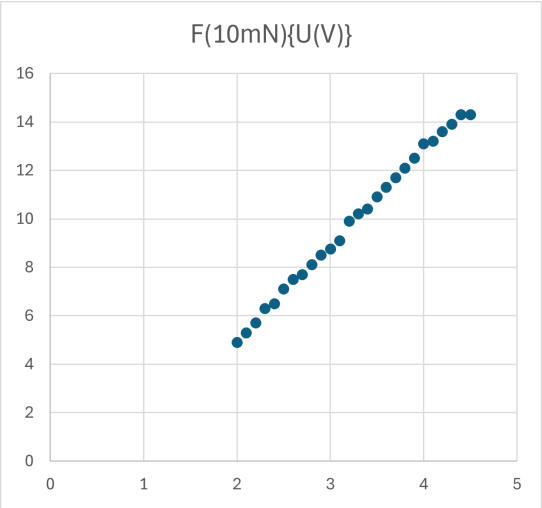
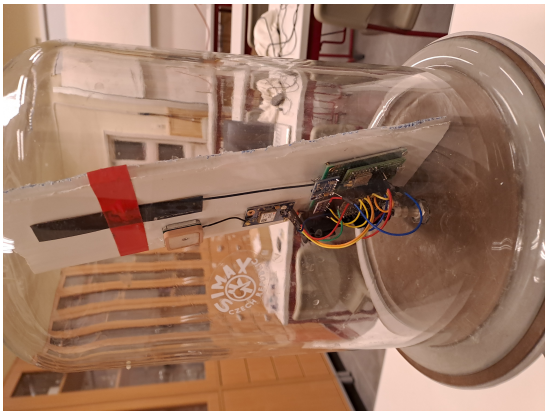


Figure 6: GNSS precision test

GNSS 14 Click: This is our GNSS module. It uses the MIA-M10Q chip. It ensures low power consumption and high performance. We communicate with it through UART. We tested the module for accuracy. We took just under 2000 measurements with a fixed module and came to the conclusion that the spread within was $\pm 2.2m$.



(a) Thrust of the motor as a function of voltage

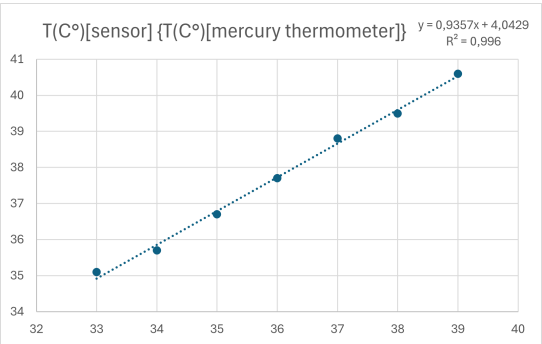


(b) The vacuum chamber used for calibrating the pressure sensor

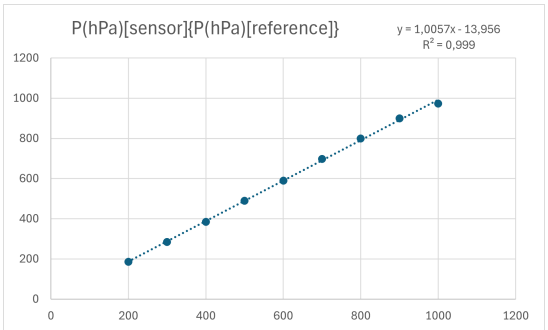
Figure 7

Motors: We choose three 6x15mm $22000KV(rpm/V)$ brushed FPV drone motors. They are optimal because of their small footprint and comparatively high thrust. We decided to drive them directly from the battery. For controlling them from the GIPO ports of the RPI and a simple MOSFET driver circuit. Our MOSFET of choice is the IRLZ44N. It is an N-channel logic-level MOSFET designed for low-voltage switching applications. It features a low gate drive voltage, thus the maximal GIPO voltage ($3.3V$) can fully turn it on. We measured the thrust of the motors with the use of a precision scale for reference.

BME280: This sensor is responsible for measuring temperature, pressure and humidity. We calibrated the sensors usings as shown below. The offset calculated from the linear regression will be used, for correction of the measured data.



(a) Temperature



(b) Pressure

Figure 8: Calibration diagrams of the temperature and pressure measurements

GY-87: This sensor consists of three different chips: an MPU6050 which is a 3 axis accelerometer and 3 axis gyroscope, a QMC5883L which is a 3 axis magnetometer and a BMP180 which is a pressure sensor. We measure the angular velocity and acceleration vectors. This data is quintessential for operating the control algorithm. Unfortunately these measurements each have significant noise. We explain how we can still get accurate data in the software design section in greater detail.

WLR089: This is the radio module we received from the CanSat team. It's a low power sub-GHz LoRa module. We use UART to communicate with the radio module. We have tested this using a Yagi type Carant Aby9 antenna connected to a laptop as our ground station. During the tests we used the $868MHz$ frequency with the highest allowed power level(PS=20, PA=ON). Our testing showed that at $1.7km$ we had an appropriate radio signal.

NCR18650B-NOPROT: This is the battery we use. It's a lithium-ion 18650 battery with a nominal voltage of $3.7V$ and a nominal charge of $3300mAh$. Even though it lacks protection this is not a problem for us because of our electrical design.

TP4056-1A-MUPR: This is our battery charger. It can charge through micro USB with the current of $1A$. It has DW01A protection and uses the TP4056 charge controller.

2.3.2 Testing the electronics

First we give a brief estimate of the power consumption according to online datasheets.

Current (mA)	Voltage (V)	Power (W)	Power consumption references	Overall current
90mA	3,7V	333mW	Raspberry Pi Pico	221,9mA
3,9mA	3,3V	12,87mW	GY-87	Overall power consumption
115mA	3,3V	379,5mW	WLR089	768,27mW
12,9mA	3,3V	42,57mW	GNSS 14 Click	
0,1mA	3,3V	0,33mW	BME280	

Table 1: Power consumption estimates

We managed to build the whole electric circuit as it is on a breadboard setup. We used it to see how everything works and to give a more accurate estimate of the power consumption and the runtime.

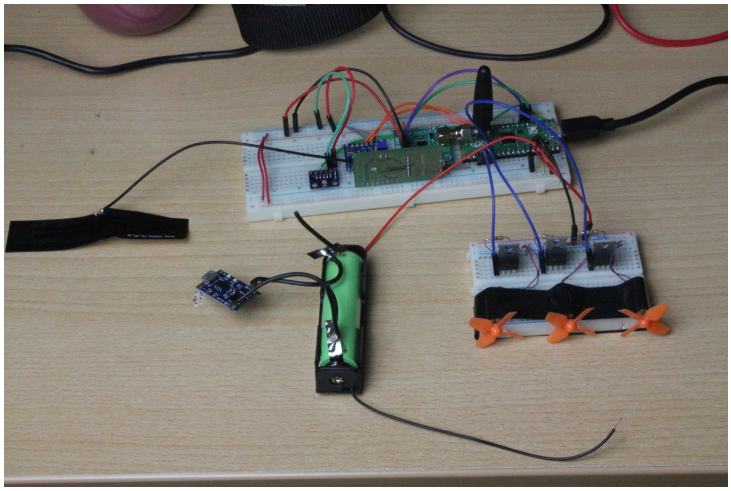


Figure 9: Caption for the image

We have performed three separate tests with this set-up. First we uploaded the full code (discussed in greater detail in 2.4), but one, that doesn't drive the motors. In the second round we gave full thrust through the entire test. We were sending sensor data for the duration of each test, but we had no GNSS signal, since the testing was done inside. With everything working we measured the following data:

/	U_1(V)	I_1(A)	P_1(W)	U_2(V)	I_2(A)	P_2(W)
TEST1	3.6	0.13	0.47	4.1	0.13	0.53
TEST2	3.75	1.75	6.56	4.15	2.05	8.51

Table 2: Power consumption test results

In the third test we wanted to simulate the motor control. In order to do this, we programmed the microcontroller, so that the motors have full thrust at a given orientation, and power cuts down rapidly outside that region. We managed a successful test: confirming the usability of the control circuit. This area will be tested more in the future, this is discussed in 3.2.3.

According to these: the runtime with two minutes of motor usage:

$$T = 120s \ + \ \frac{3300mAh - 2050mA \cdot \frac{1}{30}h}{130mA} \approx 24h$$

This is more than enough to fulfill the requirements.

2.4 Software design

In this section we will mostly talk about the onboard software as we will talk about the ground station's software later in 2.6 Ground station. We will give a description of how the software (coded in Python and running on the MicroPython firmware) works and then detail the data handling of each component.

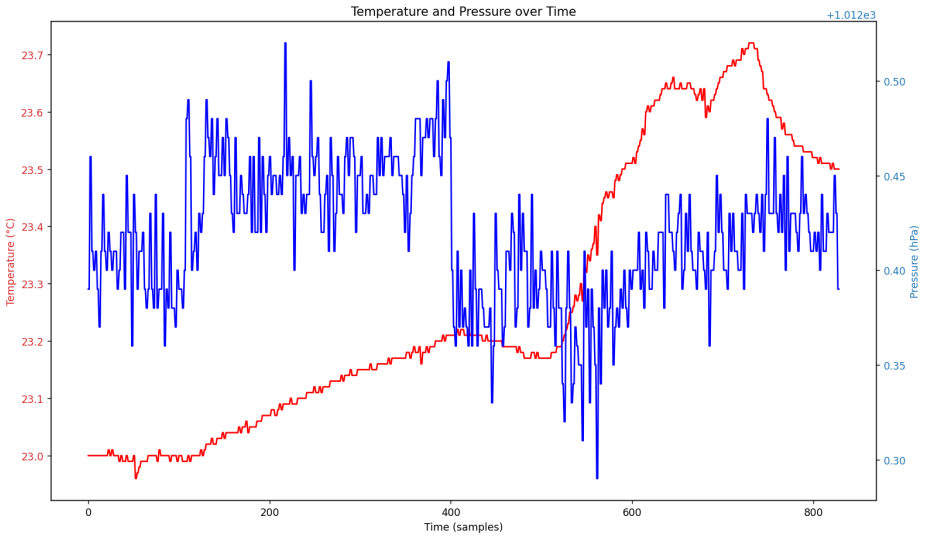


Figure 10: Temperature and pressure over time graph of BME280 sensor test

The software has several tasks it needs to accomplish, however if everything was all in one big loop there would be a significant problem. The radio and the GNSS communication both need delays both to be able to receive a response after commands are issued and to avoid spamming the UART port. On the other hand

there are parts like the navigation system where accuracy is key and as such it needs to run constantly. To address this we broke the software down into 3 parts and run these parts in parallel so that none of them take time away from the others. The first one is everything that doesn't need to run as fast as possible which also includes all the parts that need delays. The second and third are the orientation estimation and the navigation respectively, both of which need to operate with as high precision as possible. We will now detail the different threads running in parallel.

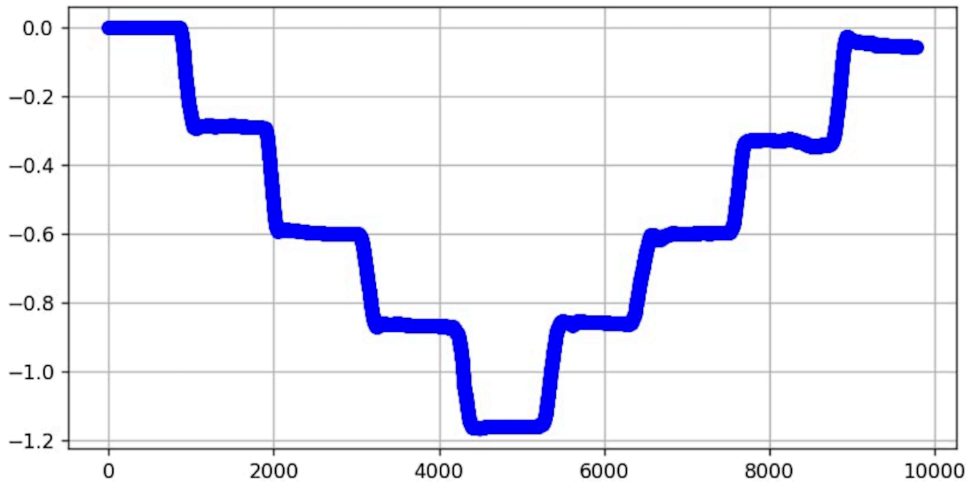


Figure 11: Yaw estimation while turning sensor by 90 degree intervals to test if it's only off by a linear scalar

- **Core 1:** Since we have 3 tasks that need to run in parallel and only 2 cores on a raspberry pi pico we run the two "fast" ones asynchronous on the same thread:
  - **Orientation:** Determining the orientation of the CanSat is crucial for navigation since it dictates which motors to activate for the correct direction. We use the GY-87 9-axis IMU sensor for this. The gyroscope alone is too unreliable due to drift, but using an AHRS (Attitude and Heading Reference Systems) algorithm called Mahony, we fuse accelerometer and magnetometer data for a more accurate estimate. Mahony determines the full orientation, though we only need the yaw, which is then passed to the navigation system.
  - **Navigation:** This part uses GNSS data, accelerometer and the yaw to compute the acceleration needed to approach the target, then applies that force by activating the motors to a specific degree. Normally, expressing a 2D vector as a combination of 2 vectors might yield negative values, but having 3 propellers over-defines the plane, so we only need to turn them one way.
- **Core 2:** This core cycles through all the "slow" tasks. Since the GNSS module is our bottleneck, producing data slower than the BME280 sensor, we start by reading the GNSS. Upon receiving a GGA-type NMEA sentence, we read the temperature and pressure, package everything into a string, and send it via LoRa to the ground station for parsing and storage. We also store the position in a thread-safe variable for the navigation system to be able to access it.

2.5 Recovery system

2.5.1 Parachute

The recovery system plays a big role in our secondary mission, so it is really important for us. Since our secondary mission is a targeted landing, our aim in connection with the parachute is to land as stable and slow as possible. For our CanSat, we chose to use a hemisphere shaped parachute, because it has the highest drag coefficient. To make it more stable, we sewed a hole into it, as it is common with parachutes. It provides a way for the air to go through, so it helps the pressure to equalize. The material we used for the parachute is made out of 100% polyester. It is ideal for a parachute, because it's light weighted and dries quickly.

Figure 12: $50N$ test of the parachute

We attached the strings to the parachute by using metal ring circles, so the material wouldn't rip. Then we attached the strings to the house of the cansat by leading the strings through the smaller holes on the house (Figure 13). We also did the $50N$ test, to see if the parachute is durable enough to put up with $5kg$ weight, and it passed the test.

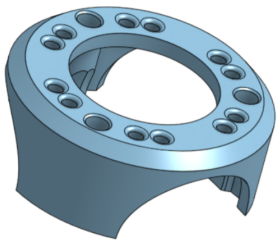


Figure 13: The top part

Figure 14: The final parachute design

We also made calculations to estimate the descent time, as required. We want to choose a parachute which allows a $5.65m/s$ descent speed to the CanSat. It would bring about a descent time of around $177s$.

We conducted a test to evaluate the performance of our parachute system. The setup involved dropping a parachute attached to a soda bottle from the 10th floor of the E building at BME. Unfortunately, during the descent, the system made contact with the building wall twice. However, after these interactions, the descent velocity remained relatively stable.

To analyze the motion, we imported the recorded video into the Tracker application and generated a position-time $(y - t)$ diagram. We calculated the average velocity, obtaining an approximate value of $7m/s$.

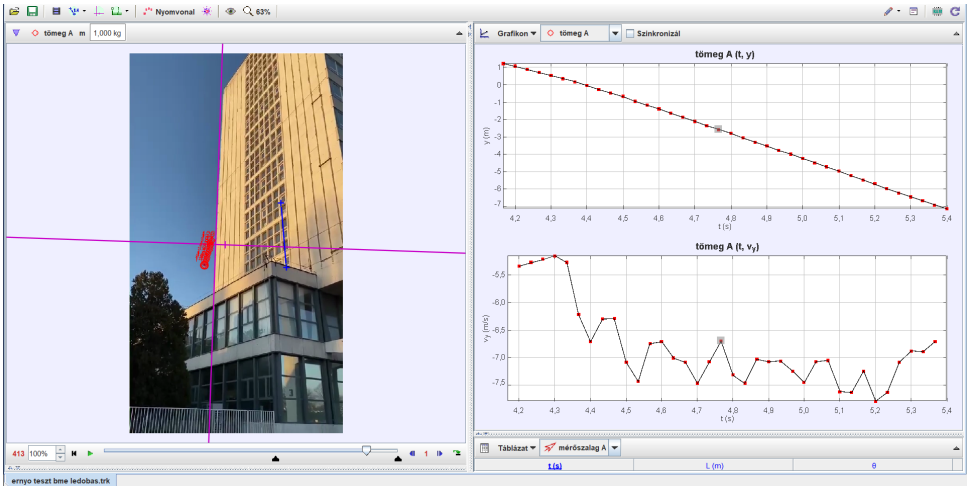


Figure 15: Drop test and evaluation

2.5.2 Guided landing

Our secondary mission is targeted landing using the propellers. We want to demonstrate the feasibility of our mission and set the parameters of our control algorithm

in the best possible way. For this purpose, we made a physical simulation of the landing.

We made the simulation in the following way. First, we approximate the rate of descent with a constant for the duration of the flight. This way we only deal with the horizontal motion. We set a target for the CanSat at $R_0 = (0, 0, 0)$, then initialize the state of the sat at $t = 0$ randomly. Then we integrate Newton's second law numerically (1) with the well-known RK4 method, giving us the trajectory of motion.

$$r(t) = \int_0^t \left(\int a(\tau)\ d\tau\ +\ v_0 \right) d\tau\ +\ r_0 \tag{1}$$

We take into consideration two forces: the air drag calculated with the well-known simple formula (2) and the control force. We simulate a slowly changing wind with the Ornstein–Uhlenbeck process (3). In the end, we plot the trajectories in 3D along with the wind speed and control force vectors. We also included a plot that shows the time evolution of the distance from the target.

$$\mathbf{F}_d = -\frac{1}{2} C_d \rho A \|\mathbf{v}\| \mathbf{v} \tag{2}$$

$$dX_t = \theta(\mu - X_t)dt + \sigma dW_t \tag{3}$$

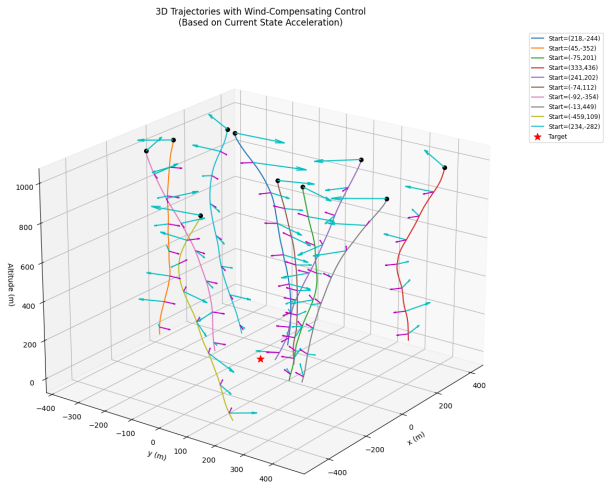


Figure 16: A 3D plot of trajectories with the inclusion of wind (blue) and thrust (purple) vectors, the red star is the target

We still need to optimize the simulation parameters, this is a preview of a test with 10 random trajectories. The control algorithm, that we used here computes the direction of the thrust as a linear combination of the desired (leaving out the wind) acceleration and the currently measured one. The simulation will be useful for analyzing the data from the flight. We're thinking about using the trajectory to estimate wind speed.

2.6 Ground station

The ground station consists of three main parts: the receiver, the local computer and the web server. The receiver consists of the same LoRa module that we use in the satellite and a yagi antenna. It communicates with the local computer through a selected COM port. We will have multiple computers with identical code if anything happens.

The module sends all data received to a C# application, which organizes the rows into a simple .txt file, handles errors and exceptions. The next step is a local Apache web server, which reads the updating data file live and transmits the data to our web server with php. The visualization consists of a height chart, position on an OSM map with leaflet.js, and a chart for temperature and atmospheric pressure.

Furthermore, it visualizes the data live, and allows us to communicate with the satellite. The primary role of this communication is to set the landing target for our secondary mission and to perform radio frequency changes. Our web server, receiving the data, performs the same visualization as the Apache server. It is publicly accessible (but obviously doesn't allow interference).

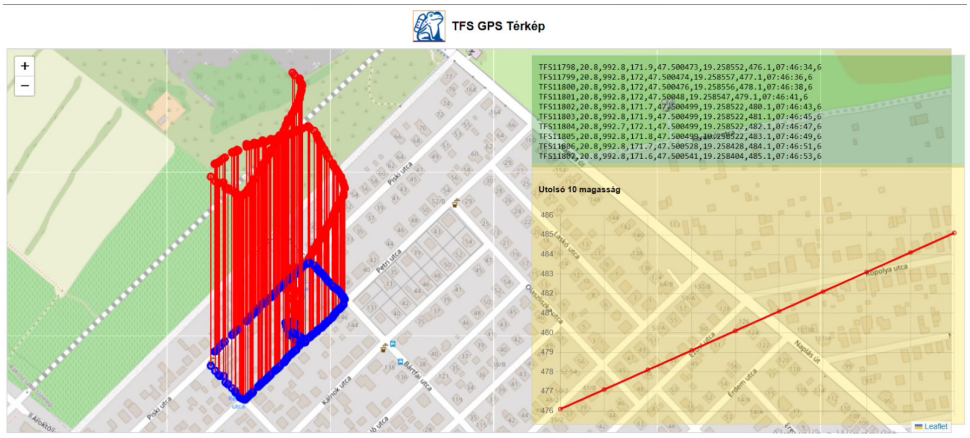


Figure 17: The current iteration of the ground station software with the data of a GNSS test

We intend to use the $868 - 868.6MHz$ frequency band and will allow for easy frequency adjustment, in the form of a command sent from the ground station, to comply with competition rules.

3 Project Planning

3.1 Time schedule of CanSat preparation

Date	Event
13 October 2024	Applied for the competition
13 November 2024	PDR (Preliminary Design Review) deadline
24 November 2024	Deadline for teammates to order all needed components for their part of the cansat
21 December 2024	Mandatory for all teammates to have a working prototype of their part
Winter Break	Testing phase: putting the cansat together and iterating on designs
Two to three weeks before CDR deadline	Started putting the CDR (Critical Design Review) together

Table 3: Time schedule in the past

Date	Event	Status
8 March 2025	Third meet up, the Cansat Gala	One cansat fully ready
24 March 2025	PLR (Preliminary Launch Review)	Everything is ready, tested, and documented
29-30 March 2025	Bajai TIE	Three cansats are ready, so we have backup
4 April 2025	The final	Presentation prepared, just need to fill in the data

Table 4: Events and important dates

Date	Event
19-23 February 2025	Ski break
21 February 2025	Ordering custom PCBs (expected arrival in about a week)
2 March 2025	Finalizing parachute parameters and the housing
5 March 2025	All parts working inside the cansat
6 March 2025	20g test
16 March 2025	Working control algorithm
21 March 2025	Put together system's test documented
29-30 March 2025	Three fully working cansats built

Table 5: Working plan for the future

3.2 Resource estimation

3.2.1 Budget

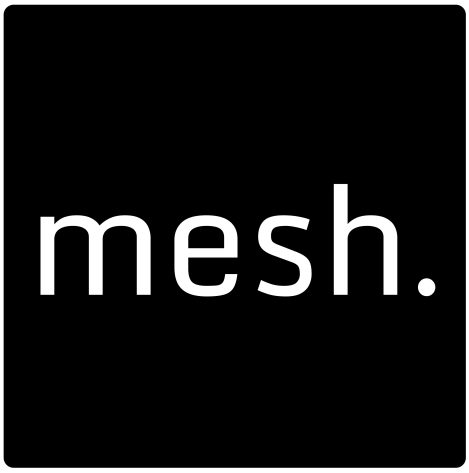
Module type	Module name	Module price (EUR)	Sum (EUR)
Microcontroller	Raspberry Pi Pico	5.54 €	136 €
Sensor (Gyroscope, accelerometer, compass)	GY-87	4.39 €	
LoRa radio module	WLR089 - CanSat	35.00 €	
GNSS module	GNSS 14 Click	34.97 €	
Humidity, temperature, pressure sensor	BME 280	3.63 €	
Battery	NCR18650B-NOPROT	6.80 €	
Cloth + thread for the parachute	-	7.36 €	
Rope for the parachute*	-	7.50 €	
Engines	6×15mm 2200KV engine 4pcs	8.63 €	
Propellers	HQ Whoop Prop 4pcs	2.95 €	
Battery charger	TP4056-1A-MUPR	0.63 €	
Transistor	IRLZ44N	1.45 €	
Heat inserts	M3 x 5.7 mm (50 pcs)	5.32 €	
Screws	M3 x 25	0.20 €	
Filament	GEMBIRD filament	12.5 €	

Table 6: Budget breakdown

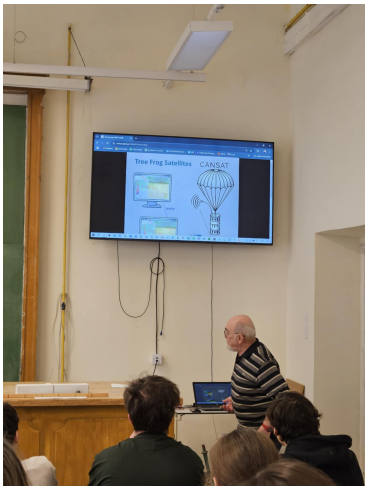
*: We got this information from the CDR recommendations

3.2.2 External support

Our main sponsor is **mesh.** (growmesh.io). They are a hackerlab, and they support motivated young researchers and engineers, just like us. They provided us office space and lab equipment during our whole project — even on busy weekends and long nights. The technical and personal insights of the mesh. team and other builders regarding the project proved to be invaluable for us. They also sponsored the sourcing of our components as well.



(a) Logo of mesh.



(b) Piláth Károly advertising the competition to students

Figure 18: Our supporters

We received a lot of support from our schools as well: Trefort Ágoston Practicing High School and Baár-Madas Reformed High School. They helped us to advertise the projects, and provided us with the physics laboratories. We made contact with Semilab as well. We can use their high tech laboratory and 3D-printers to finish up our project.

3.2.3 Test plan

We have already done a lot of testing regarding the first and secondary missions as shown above, however we still have a lot of work to do to confirm the working of all the systems. These are the following:

- Completing a working prototype of the CanSat.
- Making the 20g test happen.
- More drop tests for optimizing the parachute- and 3D design.
- Drop test with electronics, and propellers — we have to see the actual effectiveness of our return system.
- Testing the control algorithm with actual sensors and motors.

4 Outreach Program

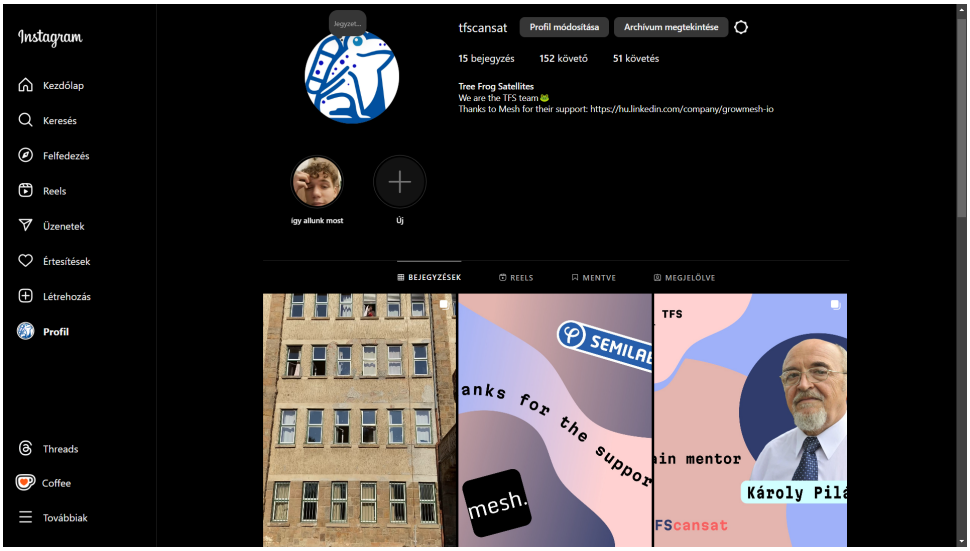


Figure 19: Our instagram page

Our main communication channel is instagram. This platform is presumably the most widely used social media site among our generation. Our goals are sharing our journey with others, introducing our team and ultimately inspiring others to start their own projects. We have a variety of instagram content:

- Introduction of the team members – post series
- Sharing our experiences from bigger events
- Educational videos detailing the electronic design of the CanSat
- Little videos in the story section – a rather unpolished, but personal insight into our work

We recently launched our website and our youtube channel to expand our online presence, so people can get to know our work in greater detail.

Figure 20: Turn the page around and you'll see us :)

Figure 21: Our stickers

As for live events: our mentor Károly Piláth advertised the project on the trefort days. One of our members (Csongi) introduced the project at the 3D design workshop held by him. We also participated at the Satellite Meeting, where we showcased our progress.

In the forthcoming months we're planning on making more content on social media, giving presentations in our school. We also designed stickers that we plan to flood the city with. (Budapest) Our website will have an english version as well.

We recorded the obligatory 5 minute video. It gives a brief overview of our process in an entertaining way. The link leading to it is attached here.

- Instagram: `https://www.instagram.com/tfscansat/`
- Youtube: `https://www.youtube.com/@TFSCanSat`
- Website: `tfs.grownresh.io`
- Email address: `tfscansat@gmail.com`
- Our video presentation: `https://youtu.be/KCw5Aq5D66s`

5 Requirements

Characteristics	Quantity (unit)	Requirement	Eligible (Yes or No)
Height of the CanSat	112.5mm	<115mm	Yes
Mass of the CanSat	244g+60g(added weight)	350g>m>300g	Yes
Diameter of the CanSat	63mm	<66mm	Yes
Length of the recovery system	30mm	<45mm	Yes
Flight time scheduled	177s	<200s	Yes
Calculated descent rate	5.65m/s	>5m/s	Yes
Radio frequency used	868–868.6 MHz	to be legal	Yes
Power consumption	130mA	Our battery has enough capacity	Yes
Power consumption (with motors)	2.05A	Na.	Na.
Total cost	136€	<500€	Yes

Table 7: Requirements and eligibility

We provide a dirve link, which contains extra or more detailed (for example the proof of the strength of the parachute) information regarding this document:

https://drive.google.com/drive/folders/1mXapk3iIDDGX-YegYCCNGtIbewLFgHl6?usp=sharing

We made a GitHub repository for the documentation of our code. A lot of code is already there, and we are working on updating it further.

https://github.com/Eagle-719/TFS_CanSat

Declaration

On behalf of the team, I confirm that our CanSat meets all the requirements set out in the official guidelines for the 2025 Hungarian CanSat competition.

Signature, place and date:

dr. Piláth Károly