MAUREL Ewan

BUT Physical Measurements
Specialization TI
IUT Paul Sabatier Toulouse

Development of a mobile system for the automation of vertical measurement profiles

Forschungszentrum, Jülich, Germany
Institute of Bio- and Geosciences 3 (IBG-3): Agrosphere
Tutor in the center :
GRAF Alexander

Internship carried out from April 7 to June 27, 2025

Follower teacher :
BRUT Aurore

ABSTRACT

Passionate about programming and looking for an internship that could fulfill my expectations, I completed my internship at the Jülich Research Center, within the IBG-3 (Agrosphere) Institute.

The main objective was to develop an automated system capable of measuring vertical profiles in the canopy of several fields using a robotic arm equipped with sensors (greenhouse gases, wind speed, etc.). I programmed the robot arm's control by creating a small application with a graphical interface. I also handled communication between the sensors and the robot, developed a script for real-time data acquisition and recording, and designed a custom 3D-printed support.

In addition, I also participated in field missions such as tree counting and collecting measurements in the fields. Tree counting consisted of measuring the height and diameter of young trees to monitor their growth. Collecting measurements in the fields represented several missions I carried out at the Selhaussen site, which belongs to ICOS. All of these will be detailed in the following sections.

This internship allowed me to strengthen my skills in programming, sensor integration, and teamwork in an international environment.

Main theme : Instrumental Computing
Key words : Sensor and Measurement Chain, Environment, Applied Computing

RÉSUMÉ

Passionné par la programmation et en recherche d'un stage pouvant comblé mes espérances, j'ai réalisé mon stage au Forschungszentrum Jülich, au sein de l'institut IBG-3 (Agrosphäre).

L'objectif principal était de développer un système automatisé capable de mesurer des profils verticaux dans la canopée de plusieurs champs à l'aide d'un bras robotisé équipé de capteurs (gaz à effet de serre, vitesse de vent, ...). J'ai programmé le pilotage du bras robot en réalisant une petite application avec une interface graphique. Je me suis aussi occupé de la communication entre les capteurs et le robot, développé un scripts pour l'acquisition et l'enregistrement des données en temps réel, et conçu un support sur mesure en impression 3D.

En plus de cela, j'ai aussi participé à des missions de terrain comme le comptage d'arbres ou de la collecte de mesures dans les champs. Le comptage d'arbre consistait à mesurer la hauteur et le diamètre de jeunes arbres pour suivre leur évolution. La collecte de mesures dans les champs représente plusieurs mission que j'ai réalisé sur le site Selhaussen qui appartient à l'ICOS. Tout sera détaillé dans les prochaines section.

Ce stage m'a permis de renforcer mes compétences en programmation, intégration de capteurs et travail en équipe dans un environnement international.

Thème principal : Informatique Instrumentale
Mots clés : Capteur et Chaîne de Mesures, Environnement, Informatique Appliquée

ACKNOLEDGEMENT

First of all, I wish to express my sincere thanks to Alexander Graf, my internship supervisor, for his support throughout this experience. He guided us from the very beginning by clearly presenting all the elements of the project and was always available to help whenever we encountered a problem.

I would also like to thank Shirin Bagheri and Sirgit Kummer for taking us into the field to carry out measurements. Many thanks to Marco Becker, Nils Becker, Odilia Esser, Antonio De Matteis and Rafaella Chiarella for their valuable assistance during the *Tree counting* sessions, alongside Shirin and Sirgit.

A thank to Normen Hermes for always delivering 3D prints on time. I thank Samuel Le Gall for taking me with him on a wheat growth measurement session in the field.

I also want to thank Arthur Saint Upery for accompanying and supporting me throughout the internship.

I am grateful to the Institute of Bio- and Geosciences 3 (IBG-3): Agrosphere at Forschungszentrum Jülich for hosting me during these three months.

I would like to thank Vincent Boulanger, who is in charge of international relations at the IUT, for his support.

Finally, many thanks to Aurore Brut, my teacher supervisor, who found this internship opportunity for me despite the difficulties we encountered, and who supervised me throughout the project work in the University Institute of Technology, helping me to prepare for this internship.

TABLE OF CONTENTS

INTRODUCTION

This report presents the 12-week internship I completed during the second year of my University Bachelor of Technology (BUT) in Physical Measurements. The internship took place from April 7 to June 27, 2025, at the IBG-3 institute (Institute of Bio- and Geosciences), within the reserch Forschungszentrum Jülich, a major German research center.

The main objective of my internship was the "integration of greenhouse gas, radiation, and meteorological sensors on a robotic arm in order to perform vertical profile measurements within the plant canopy." This project was a continuation of previous research led by Dr. Alexander Graf, aiming to analyze the exchanges of CO_2, water vapour and momentum between the soil, vegetation, and the atmosphere.

In addition to this main project, I also participated in several field activities carried out by the institute, such as maintenance of measurement stations, forest biometric campaigns (measuring tree height and diameter), and the characterization of wheat crops.

This experience allowed me to apply my scientific and technical skills in an international and interdisciplinary environment. I was able to strengthen my knowledge in instrumentation, data acquisition, and programming, while gaining insight into the operation of a large-scale research institute.

In this report, I will first present the scientific and technical background of the project, describing the measurement system developed around the robotic arm and its objectives:

- To collect measurements of CO_2, humidity, temperature, and wind speed using a robotic arm capable of performing simultaneous measurements on three different plots;
- To centralize all collected data into a single `.csv` file;
- To study the vertical dynamics of exchanges between the soil, vegetation, and atmosphere.

I will then describe the additional tasks I was involved in:

- Tree inventory on an ICOS site (height and diameter measurements);
- Analysis of young wheat shoots (growth measurements);
- Radiation measurements within a wheat field.

Finally, I will conclude by discussing what I have learned from this experience and the professional opportunities it has opened up for me.

1 PRESENTATION OF THE HOST COMPANY

1.1 Forschungszentrum Jülich

Forschungszentrum Jülich is one of the largest interdisciplinary research centers in Europe. Located near Cologne, in North Rhine-Westphalia, Germany, it employs over 7,000 staff and scientists. It is renowned for its expertise in developing scientific and technological solutions to major global challenges. The center is organized around three main research areas:

- Energy: renewable energies, hydrogen technologies, sustainable energy systems;
- Information: quantum computing, artificial intelligence, supercomputing;
- Environment: analysis and modeling of living systems, sustainable management of natural resources (soil, water, climate).

Interdisciplinarity and applied research are at the heart of its mission, in collaboration with international scientific institutions and industrial partners.

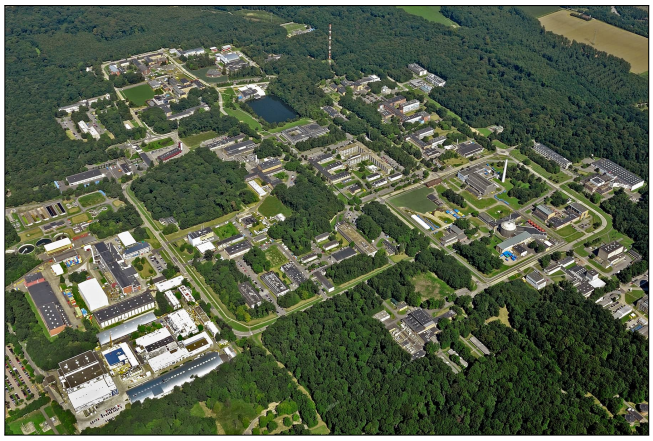


Figure 1: Aerial view of the Forschungszentrum

1.2 IBG-3 Agrosphere: Understanding Terrestrial Systems

The IBG-3 Agrosphere Institute (Institute of Bio- and Geosciences, Agrosphere) is one of the departments of Forschungszentrum Jülich. It specializes in the study of hydrological, biogeochemical, and atmospheric processes in terrestrial systems, particularly those related to agricultural and forest ecosystems.

Its main objectives are:

- To understand the interactions between soil, water, vegetation, air, and human activities;
- To model and predict the impacts of climate change, land use, and water management;
- To propose solutions for sustainable agriculture and the responsible management of natural resources.

IBG-3 relies on a wide range of scientific methods and tools:

- Ground-based measurement stations (fluxes, weather, hydrology);
- Mobile sensors or sensors deployed in the canopy;
- Remote sensing technologies and drone platforms;
- Multi-scale numerical models.
- Laboratory analyses of soil, water and plant samples

1.3 Contribution to ICOS: The European Carbon Observation Infrastructure

IBG-3 is a key contributor to the European infrastructure ICOS (Integrated Carbon Observation System), dedicated to the long-term monitoring of greenhouse gases (CO_2, CH_4, N_2O, water vapor) in terrestrial and marine ecosystems.

As part of this initiative, the institute operates in several measurement stations, three of which are fully integrated into the ICOS network:

- Selhausen (agricultural area) - we worked a lot in this site,
- Rollesbroich (grassland),
- Wüstebach (forest) — this site will be discussed later in the context of another project.

These stations are equipped with high-precision sensors measuring in particular:

- Gas fluxes between land surface and the atmosphere (eddy covariance method);
- Meteorological variables;
- Soil water and temperature conditions;
- Vegetation growth and canopy properties.

The data collected contribute to the improvement of climate models and to the development of environmental policies based on robust observations.

2 OBJECTIVES AND BEGINNING OF THE PROJECT

2.1 Main Project

The project I was involved in is a continuation of a previous study led by Dr. Alexander Graf. The initial objective of this study was to analyze CO_2, humidity, and wind fluxes at various heights under and above the canopy, using a system of mobile sensors moving vertically along a rail. (See Ney and Graf 2018 in the Web Reference)

However, this first version presented several technical and organizational limitations. On the one hand, the rail system used for vertical displacement suffered from mechanical reliability issues, which made data collection difficult. On the other hand, the sensors used operated independently, making the synchronization of measurements challenging and resulting in significant time loss during data collection.

To address these issues, a new project was launched. That is why, in collaboration with Arthur Saint-Upery, I was brought in. The objective was to design an improved version of the system based on the use of a robotic arm from the brand Doosan, model **M0617**, offering greater movement flexibility and better sensor integration.

Figure 2: Doosan M0617 robotic arm used for measurements

The Doosan M0617 is a collaborative robot with six degrees of freedom, capable of performing complex movements with high precision. Its graphical interface allows the creation of motion sequences using simple command blocks, which enabled us to quickly develop an initial vertical movement program simulating the up-and-down motion of the original rail system. We then rapidly created a more advanced program allowing the user to select the number of plots to be measured. See flowchart in Appendix 7.1. "Move J" moves the robot to a postion defined by the angles of its 6 joints (j1 to j6); "Move L" moves the robot to a position in Cartesian coordinates (x, y, z, rx, ry, rz).

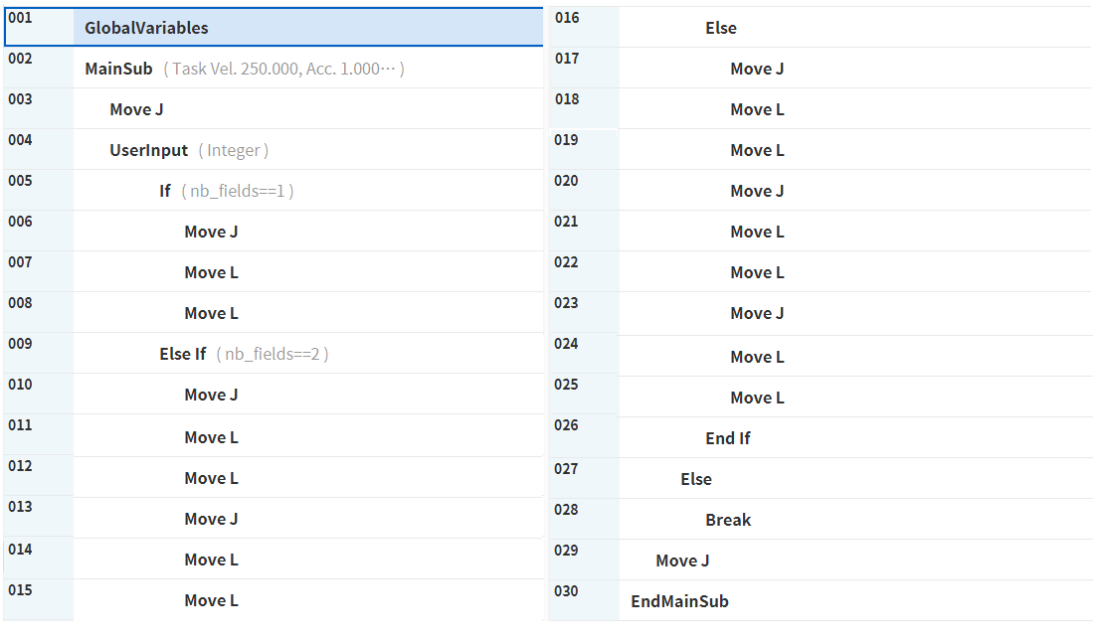


Figure 3: Excerpt of the initial control program using the robot's graphical interface

Once this system was validated, our main mission was to reprogram it in Python in order to control the robot from a computer via Ethernet communication. This transition offered several advantages:

- Automated control of the robotic arm through a Python script;
- Integration of sensors measuring CO_2, relative humidity, temperature, and wind speed;
- Centralization of all data into a single `.csv` file, with full timestamping and synchronization;
- Development of a small Graphical User Interface (GUI).

This development work formed the technical core of my internship. It involved numerous testing phases, communication adjustments, and code adaptation to match the specific constraints of the sensors used. The following sections of this report will detail the final system's functionality, the integrated sensors, and the measurement results obtained.

2.2 Other missions

In parallel with the main project involving the robotic arm, I also took part in several side missions organized by the IBG-3 institute.

First, I participated in six days of forest inventory at the ICOS Wüstebach experimental site. These sessions involved measuring the height and diameter of numerous trees in order to monitor their growth and characterize the vegetation within the plot. The methods and results obtained will be discussed in more detail later in this report.

Next, I contributed to a sampling campaign on wheat crops at the ICOS Selhausen site, aimed at comparing the effects of different treatments. The goal was to observe the differences in development between the groups by collecting samples and measuring several morphological parameters of the plants.

Finally, I took part in a radiation measurement campaign, also at the ICOS Selhausen site, which involved performing measurements at 12 points distributed along 3 lines, across 5 different spots. This mission was part of a broader study on radiation distribution at the field scale. As part of this mission, we also carried out destructive surface measurements of the leaves of wheat plants.

These diverse missions enriched my experience and complemented the technical aspects of the main project. A more detailed description of each of them will be presented in the following sections.

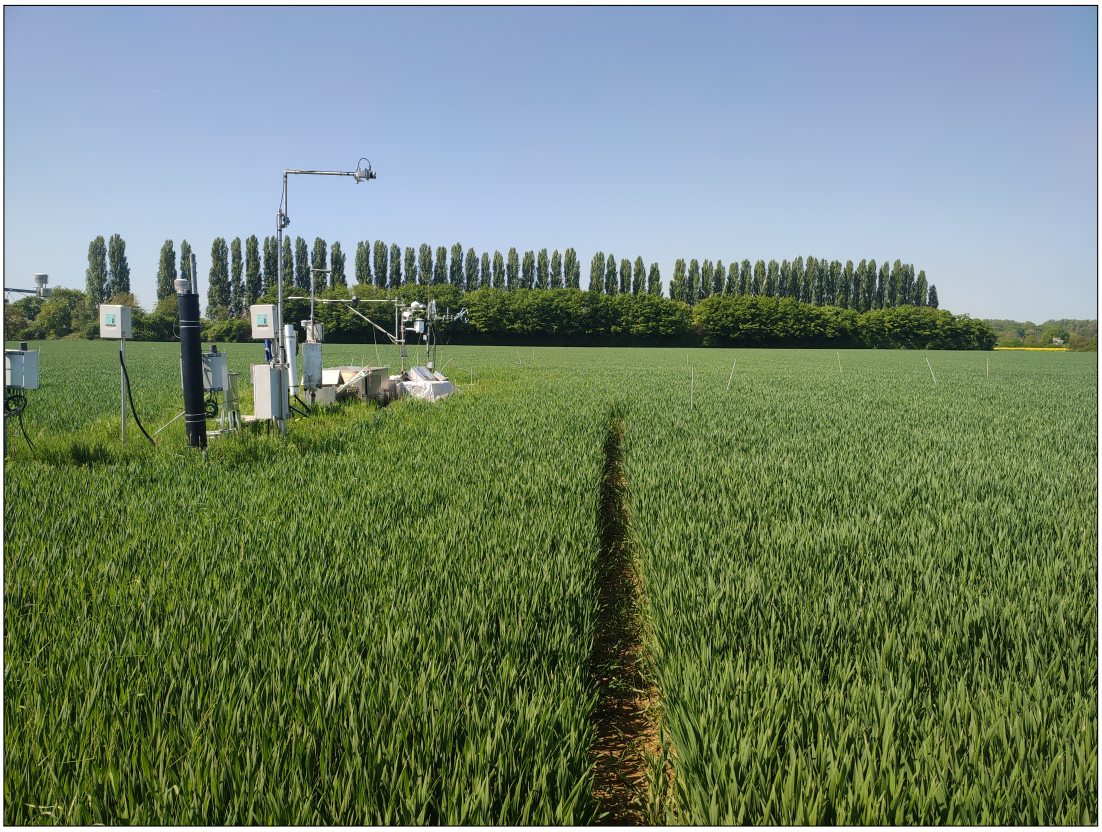


Figure 4: ICOS Selhausen site

3 Robot Arm

3.1 Tool

3.1.1 Development of the Robot Arm Control Program

After designing a first simple program that allowed the user to select one, two, or three fields to be measured, we developed a more advanced version using the robot's built-in software (The program is complicated to understand but you can find the flowchart in Appendix 7.2).

This new program offers several additional features:

- **Selection of the number of up and down movements** to perform per field — this parameter corresponds to the number of up-and-down cycles the robot must execute;
- **Selection of the number of repetitions** — how many times the entire process (X up-and-down movements over Y fields) will be executed.

The program also includes error handling to avoid invalid inputs: for example, if the selected number of fields is not 1, 2, or 3, or if the number of up-and-down cycles or repetitions is set to zero, an alert is triggered and the program stops to prevent any unexpected behavior.

After developing the program using the robot's internal software, we quickly realized that this solution had limitations. While the software allowed us to create relatively complex motion sequences, it did not address one of our key needs: centralizing both the control of the robot arm and the acquisition of sensor data within a single program.

Therefore, I began searching for a way to establish communication between Python and the robot. After some research, I found a GitHub repository that provided exactly this functionality (see reference in the Web References). The code offered a basic interface that enabled a connection between Python and the robot, along with a few simple motion commands.

Building on this resource, and using a PDF document provided by Doosan (also listed in the Web References) which detailed all the available robot commands, I started to develop a control program in Python. We hoped, switching to Python, allowed us more customizations.

First, we reproduce, in Python, the second program built using the robot's internal software. In the robot's software, we could add "Custom codes". This allows us to import 2 .txt files (available on the GitHub and wich I modified with using the Doosan PDF) that react according to the command received in the socket. With these files, we were able to control the robot with a python program (Modified files provided in my personal GitHub, listed in the Web References). Now, I will describe the different functions used in this Python program.

3.1.2 Robot Communication and Control Functions

```python
def send(self, cmd):
"""Send a command to the robot via TCP"""
  cmd += "\r"
  bytes_sent = self._socket.send(cmd.encode())
  return 0 if len(cmd) == bytes_sent else -1
```

Listing 1: send(cmd)

send(cmd): Sends a command to the robot via socket. Verifies that all bytes have been successfully transmitted.

recv(bufsize): Reads data from the socket connected to the robot. Returns the received data or "timeout" if there is no response. Useful for retrieving position data.

```python
def recv(self, bufsize=255):
"""Read data from robot TCP socket"""
  response = None
  while response is None:
     try:
       response = self._socket.recv(bufsize).decode()
     except socket.timeout:
       response = "timeout"
     except Exception as e:
       raise e
     if response:
       if "\r" in response:
          parts = response.split("\r")
          # Return the line that includes 'done' if
             found
          return next((r for r in parts if "done" in r
             ), parts[0])
       response = None
```

Listing 2: recv(255)

```python
def gotoj(self, pos):
"""Go to joint position and wait for completion
   """
  msg = "gotoj," + ",".join(map(str, pos))
  self.send(msg)
  response = None
  while response != "gotoj,done":
      response = self.recv()
```

Listing 3: gotoj(pos)

gotoj(pos): Moves the robot to a position defined by the angles of its 6 joints (j1 to j6). Blocking wait until the movement is complete.

goto(pos): Sends the robot to a position in Cartesian coordinates (x, y, z, rx, ry, rz). Blocking wait until the movement is complete.

```python
def goto(self, pos):
"""Go to Cartesian position and wait for
   completion"""
   msg = "goto," + ",".join(map(str, pos))
   self.send(msg)
   response = None
   while response != "goto,done":
       response = self.recv()
```

Listing 4: goto(pos)

These are the main basic functions. With them, we were able to recreate the same program that was originally developed on the robot's tablet.

However, by taking advantage of Python's power, we went further. The use of *threads* — functions that can run in the background while the main program continues — allowed us to run the robot control program and real-time position data collection simultaneously. This enabled us, for instance, to dynamically plot the robot arm's movement profile during measurement cycles.

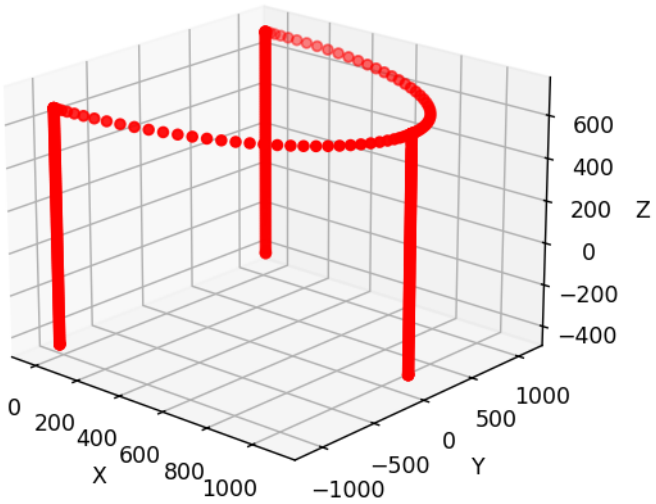


Figure 5: Example of the robot arm's movement profile during a data acquisition

After that, we developed a graphical user interface (GUI) to make the program easier to use, without the need to modify the source code.

To simplify code organization and facilitate maintenance, the program was split into three files:

- A file dedicated to the graphical user interface management (`gui.py`);
- A file for communication with the robot arm and data processing (`robot-control.py`);
- A main file that controls the overall operation (`main-program.py`).

The programs are available on my GitHub.

With this solid foundation in place, I was able to further develop the program, making it more flexible and suited to a variety of experimental conditions.

The user can now configure many parameters via the graphical interface:

- The **number of plots** to measure;
- The **orientation of each plot**, expressed in degrees relative to the robot's axis;
- The **distance of each plot** from the robot (in mm);
- The **measurement mode**:
  - In **continuous mode**, the user specifies a sampling frequency (in Hz), and the robot performs back-and-forth movements while collecting data continuously;
  - In **discontinuous mode**, the user defines a number of stop points for each ascent and descent; the robot stops at each level to perform a measurement.
- The **number of up/down cycles** per plot;
- The **number of repetitions** of these cycles across all plots.

Finally, the user can select the environmental parameters to be measured from the following:

- **Wind** (speed for each direction; U, V, and W);
- **CO_2**;
- **Temperature**;
- **Intern Pressure and Temperature of the CO_2 sensor (for correction of measurements)**;
- **Relative humidity**.

All these variables can be measured simultaneously or only those of interest (for example, wind only).

In FIGURE 6 you see the graphical interface. When you open the program and select, for exemple, 3 fields. By clicking on "Find Ground", you can manually lower the robot close to the ground. We wanted to do it automatically with contact detection (it worked) but it's not good for the sensors to touch the ground a lot of times. When the ground of each field is found, you can click on a start button, that appears, to start the acquisition. (All the different stages are available in Appendix 7.3).

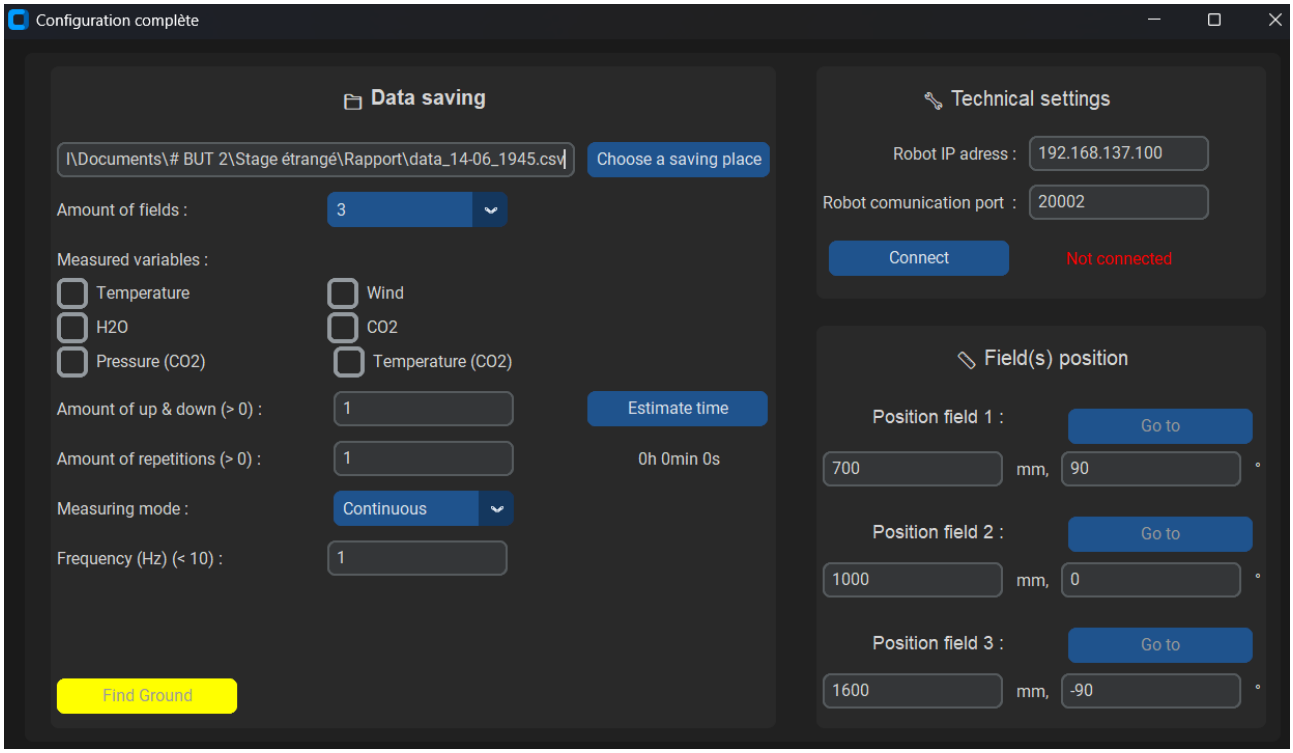


Figure 6: Graphical interface for configuring measurement parameters

3.1.3 Wind Measurement: TriSonica Mini LI-550 Sensor

As previously mentioned, it is possible to measure wind. To do so, we use the **TriSonica Mini LI-550** sensor, a compact and lightweight ultrasonic anemometer.

This anemometer measures wind speed by sending ultrasonic pulses between several sensors arranged in a cross pattern. The wind changes the flight time of the ultrasound waves : they

travel faster upwind and slower downwind. By comparing the travel times in different directions, the device calculates wind speed and direction, without any moving parts.

Its main features include:

- **Wind measurements** on 3 axes (speed range from 0 to 50 m/s, resolution 0.01 m/s, accuracy up to ±0.2 m/s);
- **External temperature** (accuracy up to ±1.5 °C);
- **Digital output** via `RS-232`, `RS-422`, or `UART`;
- **Sampling frequency** adjustable up to 40 Hz.

Figure 7: TriSonica Mini LI-550 sensor used for wind measurement

The sensor communicates with a computer via serial communication on a COM port. This type of communication allows data to be exchanged character by character at a defined baud rate. A USB-to-serial cable or a suitable converter is sufficient, along with the correct connection port. Once the data is received, it can be saved to a .csv file.

In Python, this communication is easily managed using the `pyserial` module. Below is a script that collects and displays the raw data sent by the sensor:

```python
import serial

ser = serial.Serial(port='COM3', baudrate=115200, timeout=1)
print("Connecting to the sensor...")

try:
    while True:
        data = ser.readline().decode('utf-8').strip()
        if data:
            print("Data received:", data)
except KeyboardInterrupt:
    print("Stopping program.")
finally:
    ser.close()
```

Listing 5: Code snippet to collect wind sensor data

3.1.4 Measurement of CO_2, Humidity and Pressure: LI-7000 Sensor

As previously described, we can also measure carbon dioxide (CO_2), humidity and pressure. For this, we use the **LI-7000** analyzer from LI-COR, a highly accurate and reliable instrument for continuous measurement of CO_2 and water vapor concentrations in the air.

The LI-7000 is a non-dispersive infrared (NDIR) gas analyzer. As can be seen in the diagram below, by measuring the absorption of infrared light we can find the concentrations. That provides real-time measurements with high accuracy:

- CO_2 measurement range: 0 to 3000 ppm, with typical accuracy of ±1 ppm;
- Simultaneous humidity measurement from 0 to 70 mmol/mol, with high sensitivity (<0.02 µmol/mol H_2O);
- Measurement of internal temperature and pressure for data correction;
- Fast response time, suitable for dynamic measurements in atmospheric profiles.

Figure 8: LI-7000 analyzer for CO_2 and humidity measurements

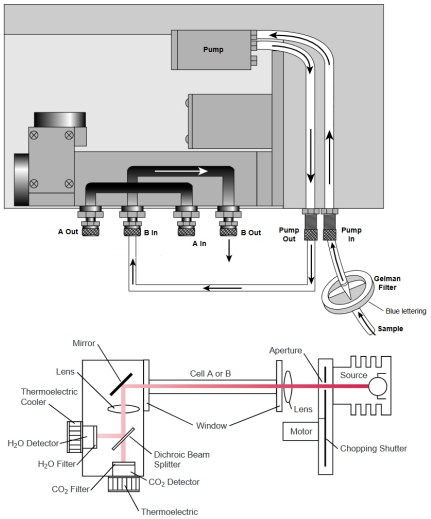


Figure 9: Interior diagram of the LI-7000

The LI-7000 communicates with a computer via a USB interface. Unlike standard serial communication, the USB interface uses specific identifiers: `idVendor` and `idProduct`, which allow the operating system to recognize the connected device.

To communicate with the sensor in Python, we use the `libusb` library, which provides low-level access to USB devices. This library allows us to detect, open, and exchange data with the sensor using its identifiers.

Quick definition of terms:

- `idVendor`: unique identifier of the device manufacturer;
- `idProduct`: unique identifier of the product provided by the manufacturer.

Below is a simplified example of initializing communication with the LI-7000 using `libusb` in Python:

```python
import usb.core
import usb.util
import time

dev = usb.core.find(idVendor=0x1509, idProduct=0x0A02)

if dev is None:
    raise ValueError('Sensor not founded')

dev.set_configuration()


data_complete = b""
try:
    while True:
        try:
            if dev == None:
                raise ValueError("error")
            data = dev.read(0x86, 64, timeout=5000)
            data_complete += bytes(data)
            while b'\n' in data_complete:
                line, data_complete = data_complete.split(b'\n', 1)

                line_list = line.decode('utf-8').split('\t')
                try:
                    data = ["CO2A", line_list[6], "CO2B", line_list
                        [8], "deltaCO2", line_list[9], "H2OA",
                        line_list[15], "H2OB", line_list[18], "deltaCO2
                        ", line_list[19], "P", line_list[21], "T",
                        line_list[24]]
                except:
                    data = []
                print(f"Receved data : {data}")

        except (usb.core.USBError, ValueError) as e:
            print("Sensor unconnected")
except KeyboardInterrupt:
    print("\nStopping program.")
finally:
    usb.util.release_interface(dev, 0)
```

Listing 6: USB initialization example using libusb

Once connected, commands can be sent and data read from the sensor using its USB protocol. This enables real-time acquisition of CO_2, humidity, pressure, and temperature values.

This type of communication is slightly more complex than simple serial communication, but it offers a more robust and faster interface, ideal for the precise and continuous measurements required in our experiment.

3.1.5 3D Mounting for Sensor Integration

To attach all the sensors together securely, we designed a custom part using 3D modeling software (Fusion 360). This part was then printed using a 3D printer. The final result allowed us to fix the sensors together in a compact and stable configuration, making them easy to mount on the robotic arm.

Below are three views illustrating the design and integration:

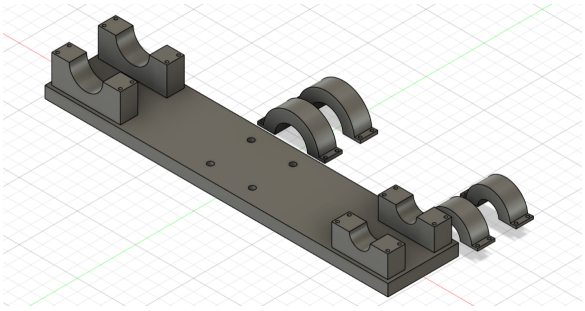


Figure 10: 3D model of the custom mount

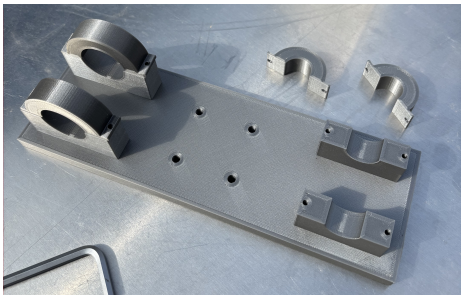


Figure 11: Printed 3D part

Figure 12: Sensors mounted on the robot

3.2 Result Measures

Thanks to good time management, we completed the development of the robotic arm earlier than expected. This allowed us to carry out a few measurement sessions with it at the end of the internship.

To gain some additional height for our vertical profiles, we applied a small workaround. Due to the arm's limited reach, a purely vertical movement does not allow sufficient coverage. However, since parameters such as CO_2, humidity, wind speed, and temperature vary much more significantly in the vertical direction than horizontally, we assumed approximate horizontal homogeneity at heights well above the canopy.

So we have vertical movement into and slightly above the field, and, once the arm is fully extended, it does not stop but continues with a parabolic upward movement. This method increases the maximum measurement height. The figure below illustrates this motion:

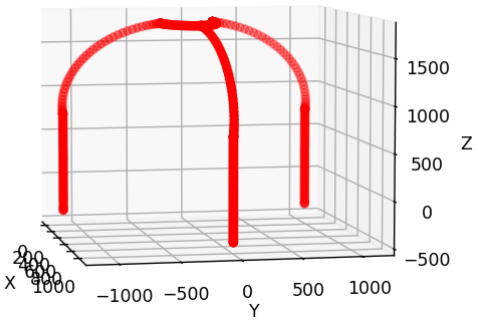
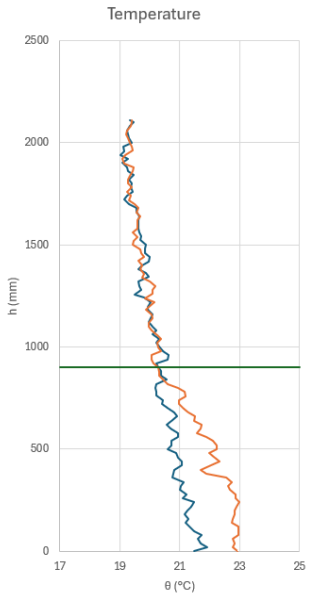


Figure 13: Arm movement : linear then parabolic

We managed to record some measurement profiles but not enought. We did not processed thoroughly the data (e.g., no standard deviation in my graphs), we calculated simple averages to give a preliminary overview. We plan to collect more data during the last two weeks of the internship. If you are interested, reference data from Ney and Graf (2018) are available online (see Web Reference).

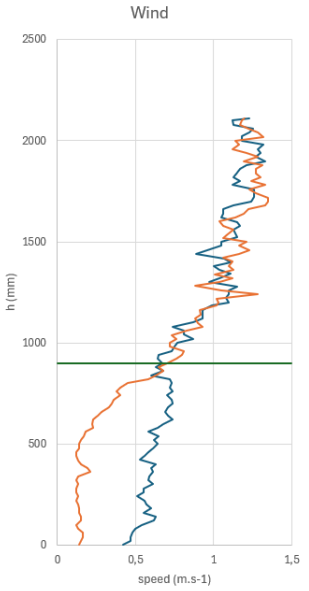
3.2.1 Temperature

There is little difference between the temperature profile above the short-cut grass reference area (in blue) and above the wheat field (in orange). The main difference is at ground level, where the wheat field is about 1.5°C warmer. This is likely due to the wheat retaining heat from solar radiation. The green horizontal line shows the height of the wheat plants.



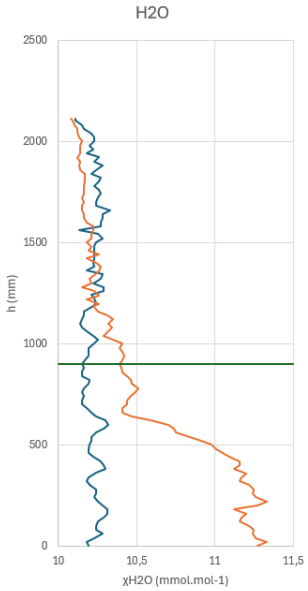
3.2.2 Wind Speed

This parameter proved more insightful. Outside the field, the wind profile shows a typical logarithmic shape: wind speed decreases toward zero near the ground. Inside the field, we first observe a near-zero linear segment corresponding to the dense crop layer, followed by the expected logarithmic increase above the canopy.



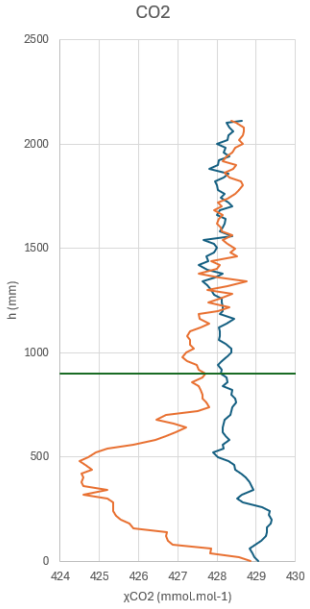
3.2.3 Humidity

Outside the crop field, humidity remains relatively constant with height. Inside the field, the profile is less clear, but we can tentatively identify two "dips" (or local minima): one related to plant transpiration and another corresponding to soil evaporation, trapped and preserved by the plant canopy. But, in our current data, it doesn't necessarily show, due to mixing.



3.2.4 CO_2 Concentration

This was perhaps the most interesting result. Outside the field, we observe a slight increase in CO_2 concentration near the ground—likely due to soil respiration—followed by a stable value in the ambient air. Inside the wheat field, we again see this soil respiration near the base. Then, CO_2 concentration drops rapidly to a minimum caused by plant photosynthesis, before rising again to match the ambient concentration above the canopy.



4 Additional Missions

4.1 Tree Counting

4.1.1 Method

We worked on the ICOS Wüstebach site, a young forest located within the Eifel Nature Park. This forest has a specific area protected from wild animals (such as deer and wild boars) by a fence forming an enclosure. This setup allows studying the impact of wildlife on the growth of young trees.

The monitoring, called Tree Counting, consists of regularly measuring the height and diameter of young shoots. The method varies depending on the tree size:

- If the tree is less than 1.30 meters tall, the diameter is measured at 10 cm above the ground.
- If the tree is taller than 1.30 meters, the diameter is measured at 1.30 meters above the ground, following the DBH (Diameter at Breast Height) standard.

Figure 14: Measuring the height of a young tree

Figure 15: Measuring the diameter of a young tree

Before our arrival, two measurement days had already been conducted. We then took part in six new sessions, generally organized with two teams of three people: one team working inside the enclosure, and the other outside.

Each young tree is tagged with a unique number. During each session, we record this number and note the two measurements: diameter and height, in a data sheet.

Figure 16: Example of an identification tag attached to a tree

When a tree could not be found, we used a GPS device containing the exact coordinates of all recorded trees. This allowed us to efficiently locate missing trees.

Figure 17: GPS displaying geolocated points of trees

If a new tree was discovered (with a height of at least 30 cm), we added it to the database and recorded its position in the GPS.

Figure 18: Recording a new tree on the GPS

Thanks to this rigorous protocol, it is possible to compare tree growth based on:

- The species type (some being more favored by animals than others);
- Their position relative to the enclosure (protected or exposed to wildlife).

4.1.2 Results

To avoid measuring all the trees, a rule was established to measure only trees within 10 meters of the fence, both inside and outside the enclosure. This creates a 20-meter-wide band forming a rectangular area with thousands of recorded trees.

The image below shows this area; for clarity, only the trees we added this year are displayed. The black line represents the fence.

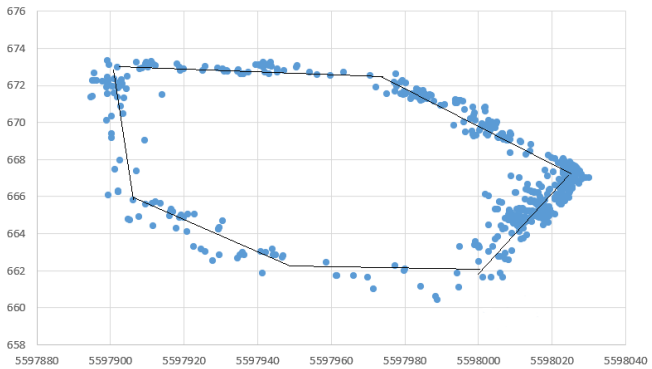


Figure 19: Tree added this year

Trees located outside the enclosure are exposed to local wildlife (such as deer and wild boar), unlike those inside, which are protected by a fence. This difference is clearly reflected in the results: there is a significant gap between the average heights of protected saplings (solid lines) and unprotected ones (dashed lines).

This gap is especially noticeable for species that remain small for several years, as they are frequently browsed or damaged by animals. In contrast, fast-growing species like birch (Betula) show much smaller differences: their rapid growth allows them to escape animal pressure early on, resulting in similar average heights inside and outside the enclosure.

The year 2024 is the last one included in the graph, as our measurements in spring 2025 were still meant to quantify the growth from the previous year.

A noticeable dip appears for birch (Betula) in 2022. This is due to the use of old and shorter measuring poles that year, which prevented their from recording the tallest trees—mostly birches. Conversely, spruce (Picea) shows a sharp drop in 2024 because the national park administration attempted to remove all young spruces to prevent them from taking over. The low average height in 2024 is therefore based on the few, mostly small individuals that survived.

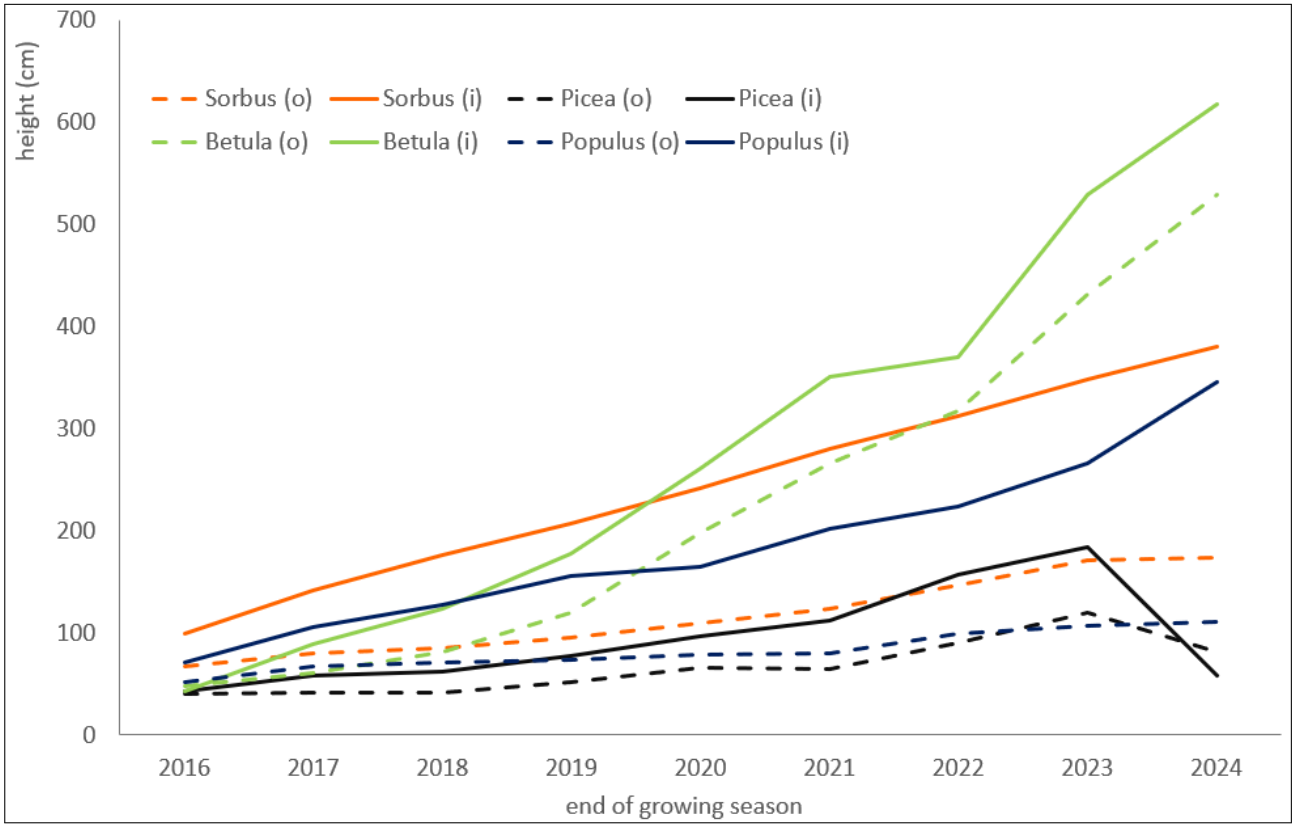


Figure 20: Evolution of tree height over the years

4.2 Other missions

Due to time constraints, I will not detail these missions and their results, but I would like to briefly mention them.

First, I participated in a mission with Samuel at the ICOS site near the robot arm station, where we collected wheat plants. The goal was to enable measurements that would help determine the optimal fertilizer concentration. The field was divided into several plots for this purpose.

Figure 21: Plot division

Figure 22: Samples in numbered bags

Another mission involved taking radiation measurements near the central measurement station of the same ICOS site. We also measured the angle between the leaves and the stem.

Figure 23: Instrument for measuring radiation and reference sensor

Figure 24: Sensor at ground level measuring the radiation

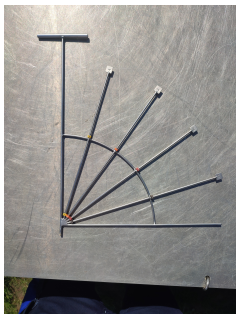


Figure 25: Tool for measuring the angle of leaves

In continuity with this study, we conducted destructive sampling to measure the total plant surface area. This involved passing all leaves and stems through a specific device that calculates surface area.

Figure 26: Plot after collecting the wheat plants

Figure 27: Sorting the different parts of wheat

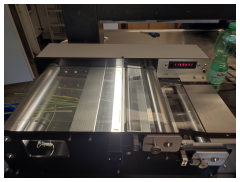


Figure 28: Measurement of the surface, here, of the stems

Finally, I was involved in a mission where we collected leaves to send to ICOS. We had to follow a very strict ICOS protocol: exactly 40 leaves per plot had to be collected using gloves, and 4 additional leaves per plot were sampled and kept moist in damp paper.

5 Conclusion

To conclude, I am proud to say that I successfully completed all the tasks assigned to me during this internship. From the development of the robotic arm control system and the integration of environmental sensors, to the participation in field campaigns, each mission was an opportunity to learn, to contribute, and to grow. I truly hope that my work was useful to my internship supervisor and to the whole IBG-3 team, and that it will serve as a valuable base for future developments—especially for the continued automation of vertical environmental profiling.

On a more personal note, I found the working atmosphere within the institute to be extremely positive and supportive. The entire team welcomed me warmly from the very beginning, and I always felt encouraged and trusted. There was a genuine spirit of cooperation and mutual help, which made the work not only productive but also enjoyable. It was inspiring to be part of a multidisciplinary and international research team, where knowledge sharing and work are truly at the heart of the activity.

This internship has had a strong impact on me. It gave me more confidence in my abilities, especially in programming, project management, and problem-solving. It also strengthened my autonomy and my ability to adapt to new challenges. I had the chance to apply many of the technical skills I acquired during my training in Physical Measurements, and I was able to see how they can be applied in a real research context. I also discovered the world of research centers and fieldwork, which was new to me, and I realized that I enjoy this type of environment, which combines technology, science, and collaboration.

Looking ahead, this experience has reinforced my desire to continue working in fields related to environmental science, instrumentation, and applied computing. It also gave me a better understanding of what it means to work in a professional and research setting, which will be very valuable as I prepare to possibly enter a work-study program next year.

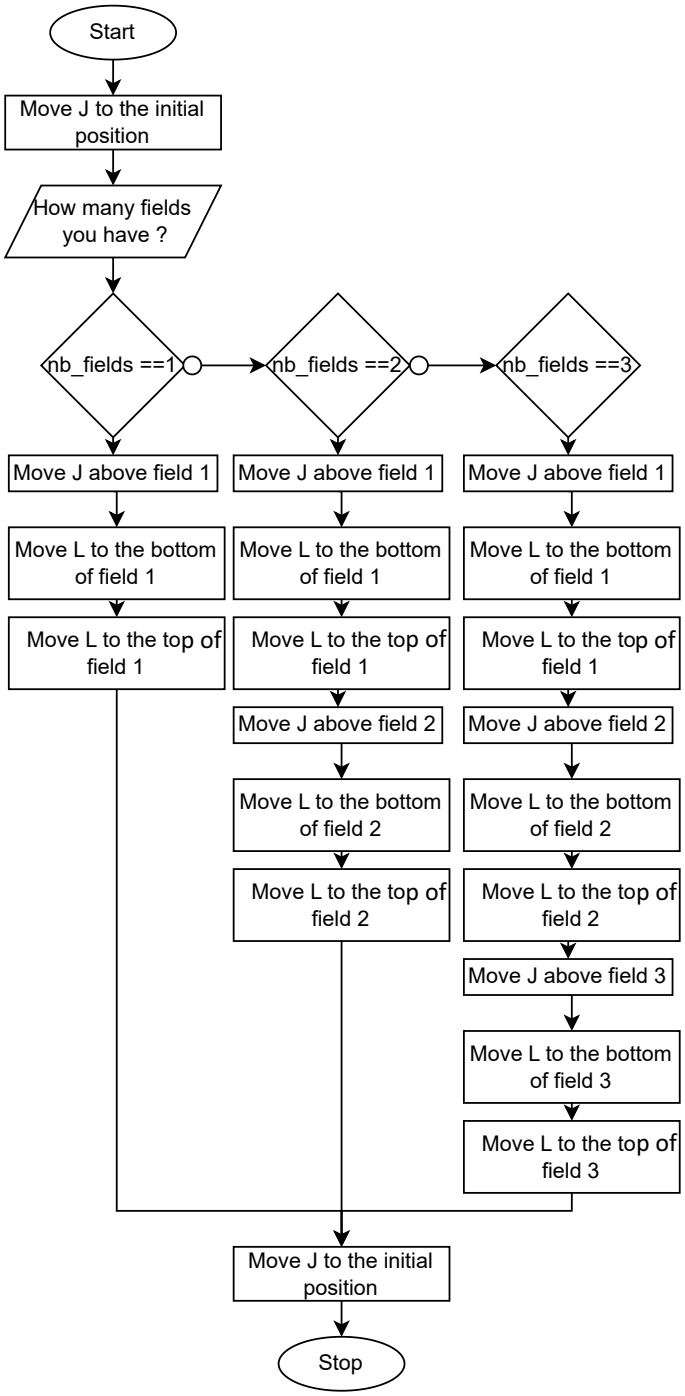
In summary, this internship has been a major step in both my academic journey and personal development. I am grateful for the trust placed in me and for the many opportunities I was given to learn and to contribute.

Web References

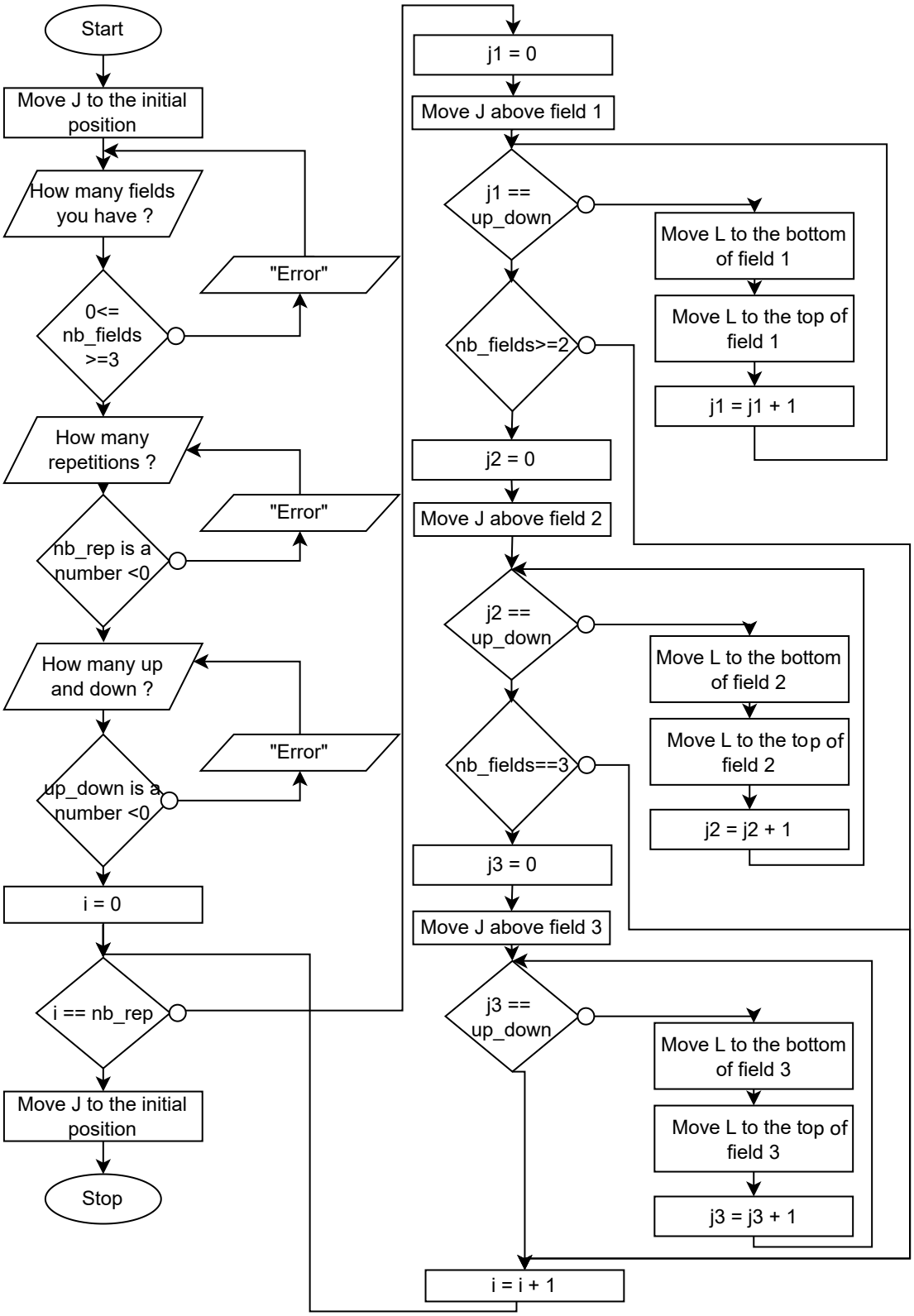
- Forschungszentrum Jülich - Main page
- Institut für Bio- und Geowissenschaften : Agrosphäre (IBG-3) - Main page
- ICOS - Main page
- Ney and Graf 2018
- Doosan Robot – Safety and User Manual
- Doosan Programming Manual (Python Integration)
- GitHub – TCP/IP Communication between Doosan and Python
- TriSonica Mini LI-550 – Product Page
- LI-7000 – Support and Manuals
- My personal GitHub - With the programs built during the internsheep

Appendices

7.1 Flowchart First Program

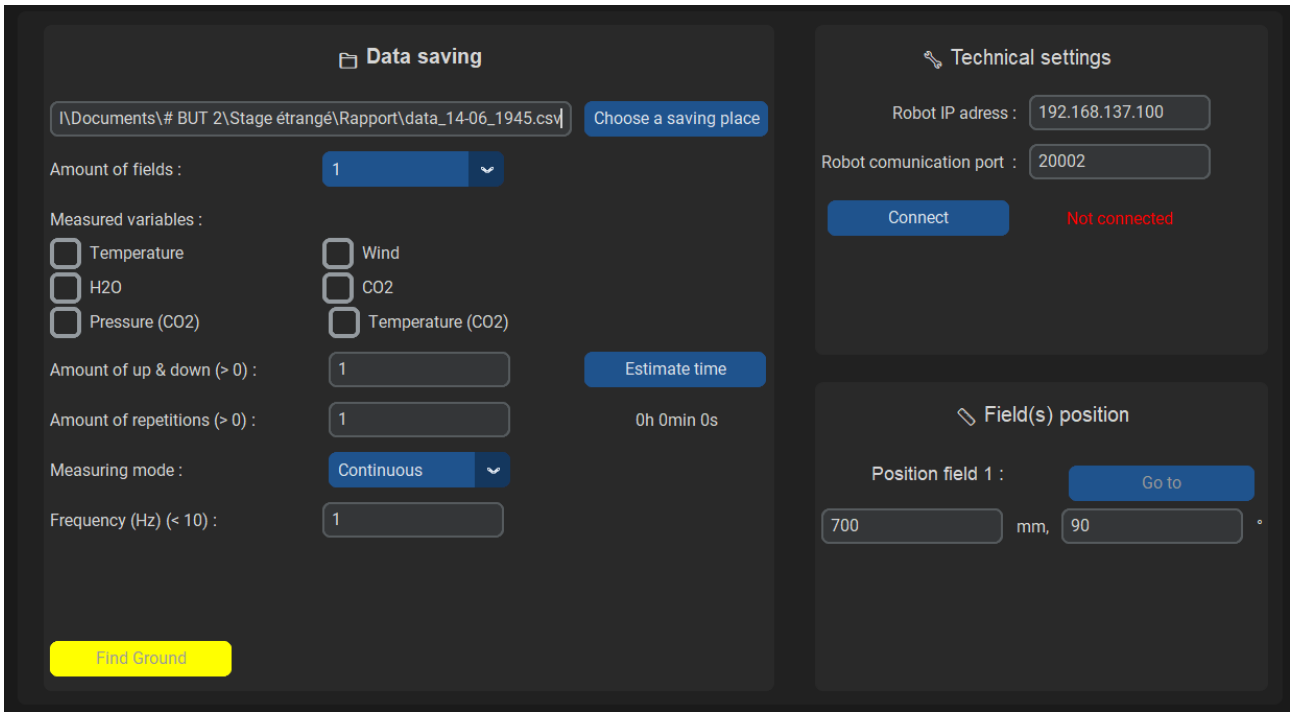


7.2 Flowchart Complete Program Doosan Software

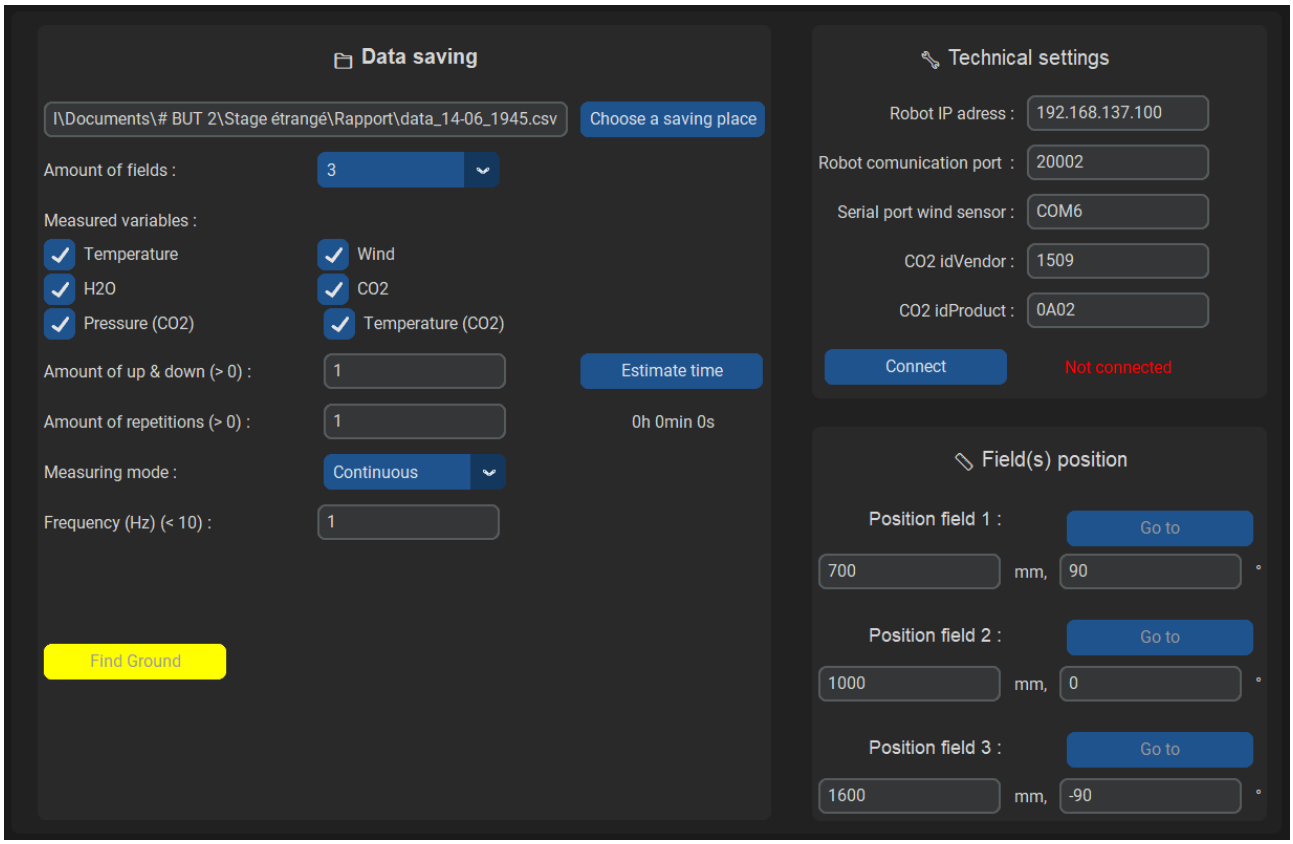


7.3 Different stages in the GUI interface

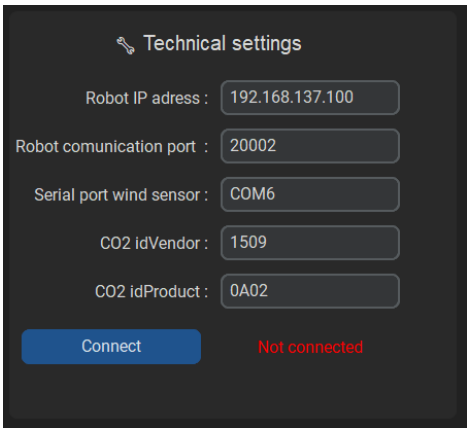
7.3.1 At the opening

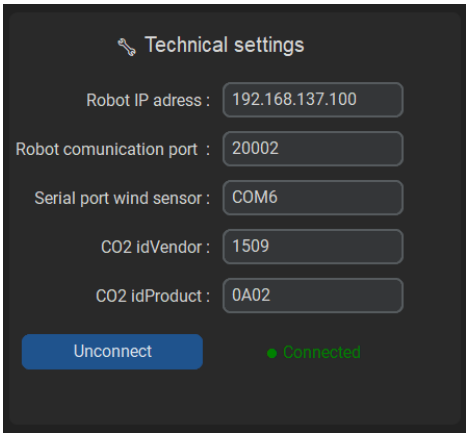


7.3.2 With all possible physical measurements and in 3 fields

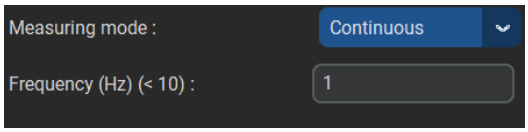


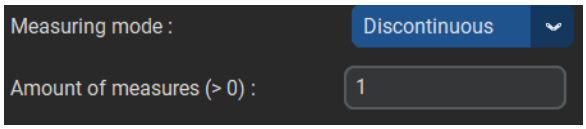
7.3.3 Connection to the Robot and to the sensors





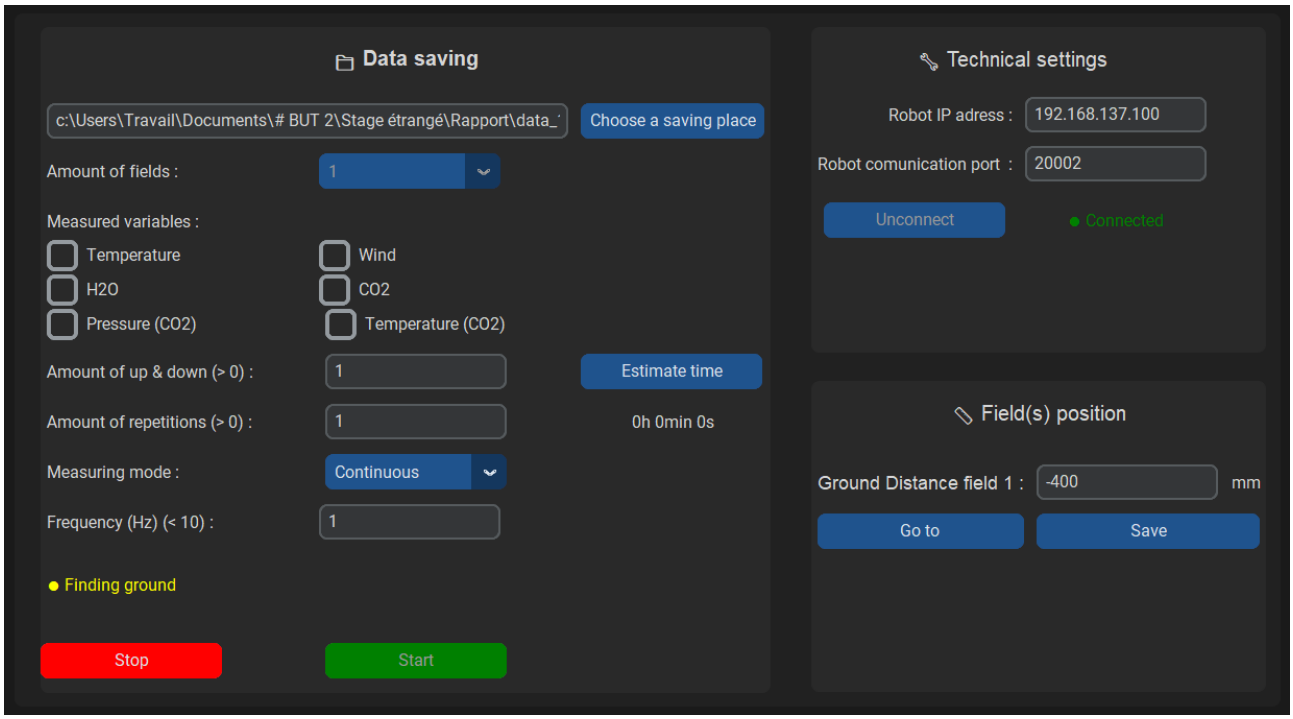
7.3.4 The two types of measurement modes



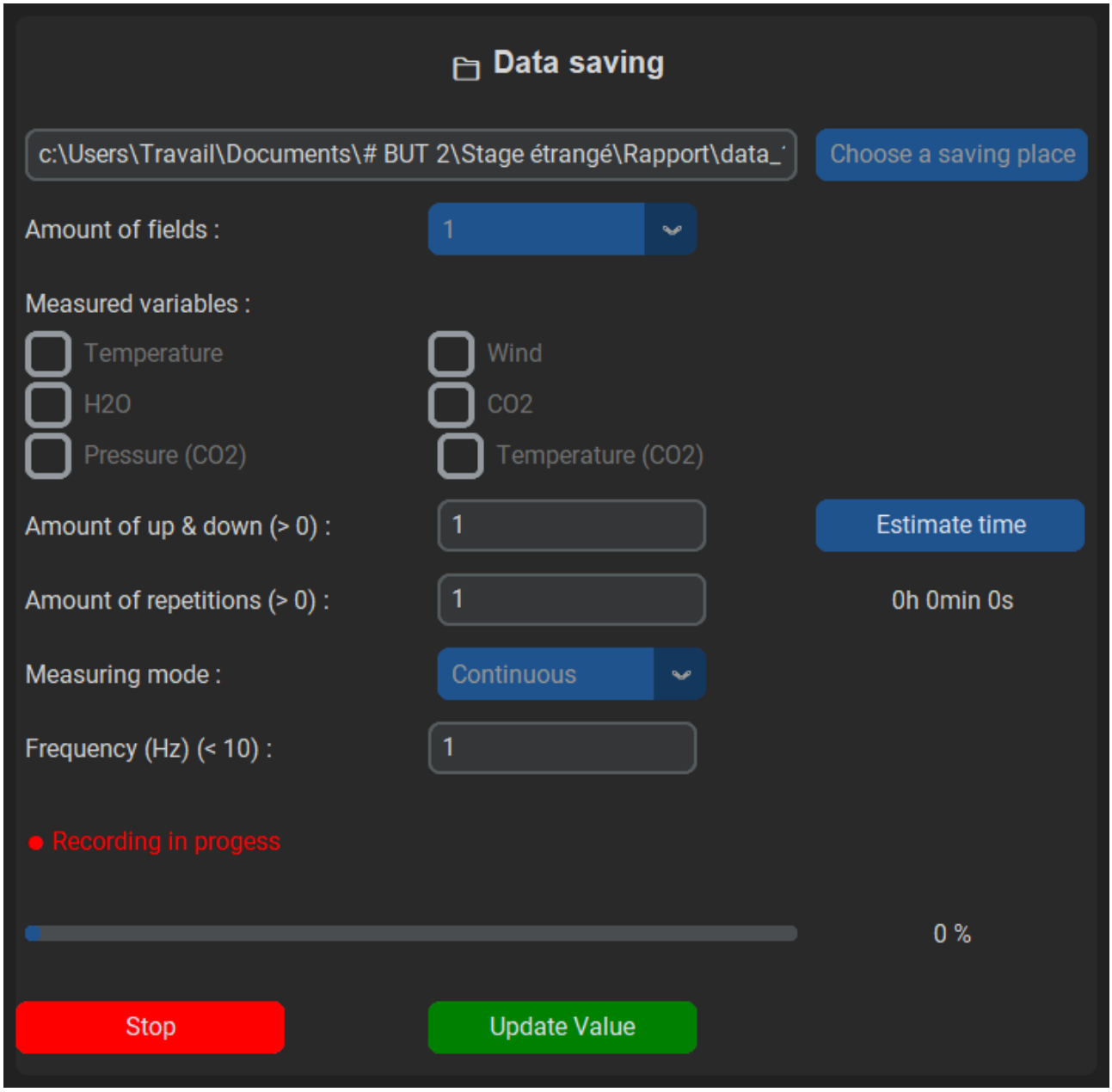


7.3.5 The button to estimate the acquisition time

7.3.6 The user enters the distance to the ground directly in the find ground mode



7.3.7 When the acquisition is launched



7.3.8 Live progress bar