FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Opportunistic Data Collection Using Drones

Vasco Tavares Carneiro

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Ana Cristina Costa Aguiar (FEUP)

Co-Supervisor: Eduardo Miguel Leitão Grifo (FhP-AICOS)

Co-Supervisor: Pedro Arlindo Vieira Madureira (FhP-AICOS)

October 16, 2019

Resumo

A tecnologia, cada vez mais presente na atualidade, desempenha um papel preponderante na evolução e desenvolvimento de novas realidades, nomeadamente na Agricultura de Precisão. A Agricultura de Precisão consiste na aplicação de várias técnicas para aumentar a produtividade e eficiência dos campos agrícolas estudando o desenvolvimento das plantas e analisando fatores que possam influenciar a produção. Um exemplo de tecnologia usada na Agricultura de Precisão é a utilização de sondas espalhadas por terrenos agrícolas, com a finalidade de medição de dados. Através da análise dos dados recolhidos é possível obter informação sobre o estado da produção e prevenir possíveis doenças que possam afetar as plantações.

Para o utilizador ter acesso aos dados, é necessário que estes sejam recolhidos e armazenados para posterior análise. O processo de recolha está, frequentemente, associado à instalação de infraestruturas para comunicação com as sondas, que, muitas vezes, apresentam elevados custos de aquisição e manutenção. Outro método igualmente utilizado é o pagamento de serviços de transferência de dados.

Esta dissertação pretende encontrar uma solução que consiste no desenvolvimento de um sistema, de fácil utilização, com a principal finalidade de recolha oportunística de dados usando um drone autónomo e focando no caso de uso da agricultura de precisão. Os dados recolhidos são armazenados e adquiridos por sondas espalhadas por um terreno agrícola, equipadas com sensores de medição de diversas grandezas físicas, como humidade relativa.

Para reduzir custos e não ser necessária a instalação de uma infraestrutura de comunicação, este sistema usa Bluetooth Low Energy (BLE) para a transmissão de dados entre equipamentos. Com base na localização geográfica das sondas, é criado um plano de voo para o drone. Com este plano de voo, o drone navega até ao ponto geográfico onde a primeira sonda se encontra e permanece a sobrevoar esse local até que a transferência de dados entre os dispositivos esteja concluída. Quando a transferência estiver concluída, o drone continua a sua rota navegando até ao próximo ponto pretendido. No fim da recolha de todos os dados presentes nas sondas, o drone regressa ao local de partida. Assim que o drone termina o seu plano de voo, todos os dados armazenados são transferidos para um computador onde são apresentados ao utilizador final permitindo uma análise mais fácil e intuitiva.

Abstract

Technology, increasingly present in everyday life, plays a significant role in the evolution and development of new realities, particularly in Precision Agriculture.

Precision Agriculture involves the application of various techniques to increase the productivity and efficiency of agricultural fields by studying plant development and analyzing factors that may influence yield. An example of technology used in Precision Farming is the use of probes spread across the field for data measurement.

Through the analysis of the collected data, it is possible to obtain information about the state of the production and to prevent possible diseases that could affect the crops.

To have access to the data, it must be collected and stored for further analysis. The collection process is often associated with the installation of infrastructures, which often have high acquisition and maintenance costs to communicate with the probes. Another used method is the payment of data transfer services.

This dissertation aims to find a solution that consists of the development of a user-friendly system, with the primary purpose of opportunistic data collection using an autonomous drone and focusing on the use case of precision agriculture. The collected data is stored and acquired by probes spread over the field, equipped with sensors to measure relevant data such as relative humidity.

To reduce costs and dismiss the installation of communication infrastructures, this system uses Bluetooth Low Energy (BLE) for data transmission between equipment.

Based on the geographical location of the probes, a drone flight plan is created. With this flight plan, the drone navigates to the geographical point where the first probe is located and remains stable in that location until data transfer between devices is complete. When the transfer is done, the drone continues its route by navigating to the next desired point. After collecting all data, the drone returns to the place of departure. As soon as the drone completes its flight plan, all stored data is transferred to a computer where it is presented to the end-user for more accessible and intuitive analysis.

Acknowledgments

I would like to thank my supervisor, Prof. Ana Aguiar, and both my co-supervisors from Fraunhofer Portugal AICOS, Engineers Eduardo Grifo and Pedro Madureira, for all their support, dedication and academic guidance during this dissertation. I also want to thank Fraunhofer Portugal AICOS and its team for the opportunity, help and support given.

I also want to thank all my closest friends who have accompanied me over these years, making this stage of my life unforgettable.

Last but not least a big thank you to all my family who supported me unconditionally. Especially to my parents, sister and brother-in-law since without them none of this would be possible.

And to all my relatives who are no longer present but would like to see me reach this point in my life.

Vasco Carneiro

“Sempre chega a hora em que descobrimos que sabíamos muito mais do que antes julgávamos.”

José Saramago

Contents

List of Figures

List of Tables

List of Abbreviations

ADB	Android Debug Bridge
API	Application Programming Interface
BLE	Bluetooth Low Energy
CPU	Central Processing Unit
ESC	Electronic Speed Control
GCS	Ground Control Software
GPIO	General Purpose Input/Output
GPRS	General Packet Radio Services
GPS	Global Positioning System
IDE	Integrated Development Environment
RPM	Revolutions per Minute
RTF	Ready to Fly
SDK	Software Development Kit
UART	Universal Asynchronous Receiver-Transmitter
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle

Chapter 1

Introduction

In recent years there has been an expansion in the development of hardware and software for UAVs, usually called drones, thus increasing their use cases. The Federal Aviation Administration (FAA) estimates that 450,000 UAVs will be flying until 2022, contrasting with 110,000 UAVs in operation at the end of 2017, which is four times larger than the previous number of UAVs [6]. This expansion has allowed the development of systems that use drones as part of the solution to the problem. UAVs are increasingly used in areas related to real-time surveillance and rescue, remote sensing, aerial photography, and transport of objects, sometimes related to health care and agriculture. These drones frequently present a decent degree of computational power, allowing, even more, the number of use cases to grow [7].

1.1 Context

The popularity of UAVs and the technology involved in their creation and use are growing recently. Due to their rapid development and popularity, drones are increasingly being found at more affordable prices. The easy access to drones allows more use cases where drones are used, taking advantage of its flexibility and the numerous sensors that can be packaged on it [8]. As an example, the scenarios for the use of UAVs in precision agriculture are frequent, mainly in the collection of images to control the growth status of plants and their evolution [9].

1.1.1 Precision Agriculture

The population has been increasing over time and getting more demanding about food quality and the impact of food production on the environment. The Food and Agriculture Organization of the United Nations (FAO) published that food production will have to grow by 70% to support global consumption until 2050 [10].

To fight against the pressure of the economy and growing population, technology joins with agriculture to develop devices and techniques for Precision Agriculture use. Precision agriculture consists in the application of several techniques, including technology, to increase the overall efficiency and productivity of a field, by studying the evolution of plantations and soil conditions, allowing the creation of alternative and favourable strategies to successful agriculture. The use of probes, distributed by agricultural fields and collecting data, is an example of technology used in Precision Agriculture. These probes collect data like temperature, soil moisture, pH, and lighting [11]. The analysis of data that is available all over the fields allows a high ability to control and manage productions. Through this data, it is possible to obtain information concerning the development, sustainability, and production of yields, preventing diseases that can affect crops [12].

1.2 Motivation

Since data analysis of agricultural fields is relevant to comprehend the state of the plantation and its evolution, the way data is collected is a significant concern [11]. One way to collect the data is to use probes equipped with sensors, which are located at specific points in the field. These probes allow the measurement of essential data for agriculture, such as soil moisture, lighting, and temperature [13]. The collection of data from probes could be a challenge because the data is spread all over the field and is required to be stored and viewed in a unique location.

In some cases, the data stored is collected manually by a person who crosses the entire field until all data is collected. This situation implies the need for a person who frequently collects the data, becoming more time consuming and inefficient.

To make data collection more efficient and without the need of a person, autonomous data collection solutions are often used. In these solutions, the probes are equipped with communication modules to make contact with each other or with a database. To be able to establish a connection between devices, it is often necessary to install infrastructures that cover the desired area, such as large antennas or wire networks. These infrastructures may incur high acquisition, installation, and maintenance costs. The probes using these infrastructures are usually equipped with large batteries and associated with high energy consumption, making them more expensive and inefficient [14].

Some probes can be equipped with GPRS communication modules, not requiring the acquisition and installation of large infrastructures. However, the use of this type of communication implies the payment of services to communication companies.

Given all these factors, the need for a cheaper and user-friendly system arises. Thus, the status of the crops can be obtained autonomously and more efficiently.

1.3 Objectives

This dissertation aims to provide a solution to the aforementioned problem by using unmanned aerial vehicles (UAVs) to collect data stored on probes previously installed in the agricultural field, acting as a means of transport for communication between the probes and the user.

Instead of using a traditional and expensive probe, where large batteries are needed, this project aims to use a cheaper and low energy consumption probe and complement its short range by using an autonomous drone to collect the stored data. The use of UAVs allows for great flexibility and mobility in data collection. The drone can go across the field, going probe to probe collecting data, through a wireless connection. After collecting all the data, the UAV returns to its base, and then it transfers the data allowing the user to visualize and consult it. Once the desired location is reached, the drone approaches the probe taking into consideration the power of its signal, moving closer to the probe for more efficient and faster transmission.

In addition, the system intends to create a flight plan for the drone, taking into account the intensity of the wind and its direction, creating a more favourable route with less energetic expenses. It also aims to ensure the return of the drone in case of low battery, changing in real-time the flight plan, if necessary.

The system can be divided into several modules that all together will allow the system to function.

- Manage the mission and the flight plan.
- Know the mission status and decide when to connect to the probe and when to resume the mission
- Manage the Bluetooth connection
- Manage the collected data

1.4 Document Structure

This document is divided into six chapters.

In chapter one is made a presentation of the project taking into account the context of Precision Agriculture, the motivation and objectives.

The second chapter of this document is devoted to the literature review. A presentation of the UAVs and UGVs is made with the advantages and disadvantages of its use in agriculture environment and also some examples. In this chapter is also introduced the available hardware and firmware possible to be used in this project.

Chapter three presents the approach considered for this project and the developed system design. The selected system components, the objectives and needs of the project are presented in more detail.

In next chapter is described the implementation and integration of the system components.

The last two chapters are dedicated to the evaluation and discussion of the system and also the conclusion and future work that can be done to increase the value of this project.

Chapter 2

Literature Review

2.1 Automation in Agriculture

A continuously search for new ways to improve efficiency in Agriculture is an increasing concern. The advancement of autonomous systems gives the opportunity to create a new range of automatic systems for agriculture use [15].

These systems range from mobile ground robots, manipulators to drones that have been tested and used to perform repetitive and standardized agriculture tasks. For a functional integration with humans and the surrounding environment, those systems are continually collecting and analyzing data being capable of working in natural conditions [16].

2.1.1 Autonomous Vehicles in Agriculture

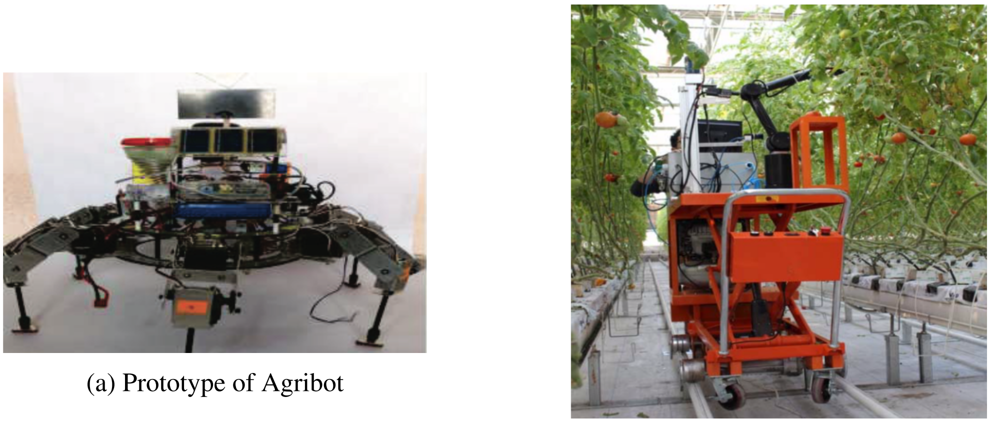
Application of autonomous vehicles in the field of machinery design and realization of agricultural tasks has been increasing. An autonomous performance for the developed systems gives the opportunity to develop a new set of tasks that can be executed according to desired standards [17]. The autonomous vehicles can be categorized into three large sections, considering their type of locomotion: the aquatic locomotion, the terrestrial locomotion, and the aerial locomotion [18]. In the agriculture environment the large majority of vehicles use terrestrial and aerial locomotion.

The terrestrial locomotion vehicles, also called unmanned ground vehicles (UGVs), will be addressed in the next subsection and the aerial locomotion vehicles, frequently called unmanned aerial vehicles (UAVs), will be addressed in the section 2.1.1.2.

2.1.1.1 Unmanned Ground Vehicles (UGVs)

Unmanned Ground Vehicles (UGVs) are usually used for several agriculture purposes. Theses vehicles can have many tasks such as: place seeds on soil, harvest plants, spray pesticides over the field, irrigate the plant, and can also monitor weather and soil conditions. For these tasks, the UGV considers the type of plant to know the nutrients and the environment that the plant needs [15].

Agribot (figure 2.1a) is an example of an autonomous robot for agricultural use. It was developed to perform all sorts of agricultural activities like drilling, seeding, weeding, and spraying fertilizers. It is equipped with several servos that allow the mobility of its legs and motors for all other actions that it can do [19]. Also, a tomato intelligent picking robot (figure 2.1b) was tested to pick-up tomatoes using a vision unit to identify the mature fruits and a jointed manipulator with 4 axis of freedom to catch them. This machine can move on pre-installed rails between plants [20]. The two systems presented illustrate use cases where UGVs perform agricultural functions, reducing the amount of human labor and the accomplishment time.



(a) Prototype of Agribot

(b) Prototype of tomatoes harvesting robot

Figure 2.1: Example of UGVs used in agriculture

The abilities of UGVs can be grouped into four categories: guidance (the way that the vehicle navigates on the field), detection (such as use vision system to identify the fruits), action (to execute tasks) and mapping (construction of field's map) [21].

UGVs have the capacity to carry more weight comparing to other vehicles. This feature is a great advantage because it allows the vehicle to be equipped with large batteries, several sensors and actuators providing a high-level of autonomy [22]. Most of these vehicles are highly robust, and it is possible to communicate with them wirelessly. With this communication, the user can be informed about their status and the status of the tasks.

Despite all these advantages, the UGVs also have some limitations concerning movement and accessibility. They have restrictions concerning some places that can be unreachable because of the rough terrain, the size of the vehicle, or the presence of obstacles on the way [23]. These vehicles also create an impact on the field, compacting the soil beneath them, which can be disadvantageous to the agricultural environment.

2.1.1.2 Unmanned Aerial Vehicles (UAVs)

The unmanned aerial vehicles, usually called drones, are a potential and increasing market. When applied in agriculture, they let the users see their fields from above. Using specially designed cameras, multispectral pictures can be taken, allowing the user to see irritation problems, soil variation, and pest infections [9]. The use of drones to spray pesticides across the agricultural field is also a use case of drones in agriculture [24].

QuestUAV drone (figure 2.2a) is a fixed-wing drone that is operated over agricultural areas and use imaging with multispectral sensors to monitor plant growth and examining crop heath. Theses drones carry different sensors such as optical, multispectral, and thermal modules to measure plant density, to detect pests and diseases, and also to identify water sources. They can cover hundreds of hectares in a single flight at a lower cost, when comparing with manned flights or ground surveys. [25]. Some drones are also used to spray liquids over the fields, as is the case of DJI MG-1S (figure 2.2b). This multirotor drone is equipped with several technologies to provide reliability during flight, spray systems and flow sensors to monitor the spraying rate in real-time. These additions make precise plant operations efficient and real-time agricultural management possible [26].

(a) QuestAUV

(b) DJI MG-1S

Figure 2.2: Example of UAVs used in agriculture

The UAVs can be divided into vehicles that use motors and control surfaces to move, also called fixed-wing drones, like the QuestUAV; and into vehicles that use just motors to move, also called multirotors, like the DJI MG-1S. The fixed-wing drones consist of a rigid wing with an airfoil that makes the flight possible by generating lift due to the drone's forward speed. An electric motor creates this speed with a propeller installed on. The control of these UAVs is guaranteed by an aileron, an elevator, and a rudder. These components allow the drone to rotate in three axes (Pitch, Roll, Yaw) that are perpendicular to each other and intersect the UAV in its center of gravity, as shown in figure 2.3 [27].

The fixed-wing drones have a more straightforward structure compared to the multirotor drones, providing less complicated maintenance and repair process. The most known characteristics of these drones are their more extended flight autonomy, their higher speed, and their greater altitude of flight. The combination of these three characteristics allows the fixed-wing drones to cover large areas in less time, increasing efficiency. Although the use of fixed-wing has many benefits, these drones also have limitations that can influence the choice of the desired platform. The fixed-wing drones need a runway or a launcher to take off and land because they can not take off and land

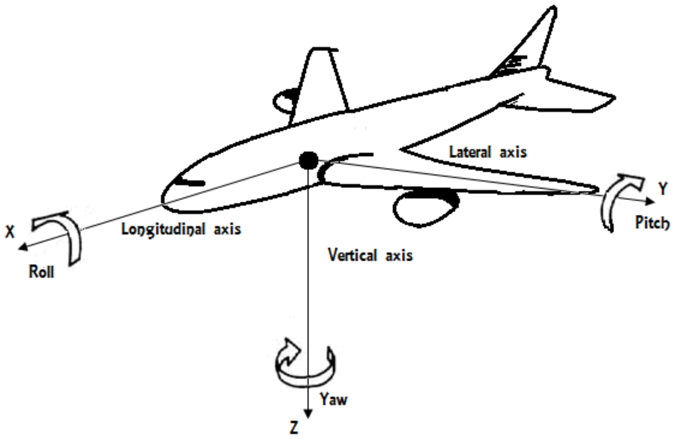


Figure 2.3: Three axes of fixed-wing drone [1]

vertically. Additionally, they can not stay in a stationary position, the same way that a multirotor drone can, during flight [28].

The multirotor drones are the most commonly used drones, and they are composed of a central body and motors with propellers installed on [29]. The majority of this type of drones can have three motors (tricopter), four motors (quadcopter), six motors (hexacopter) ou eight motors (octocopter), as can be seen in figure 2.4. The quadcopters provide the best balance between control, maneuverability, lift, and cost, being the most used and popular drones.

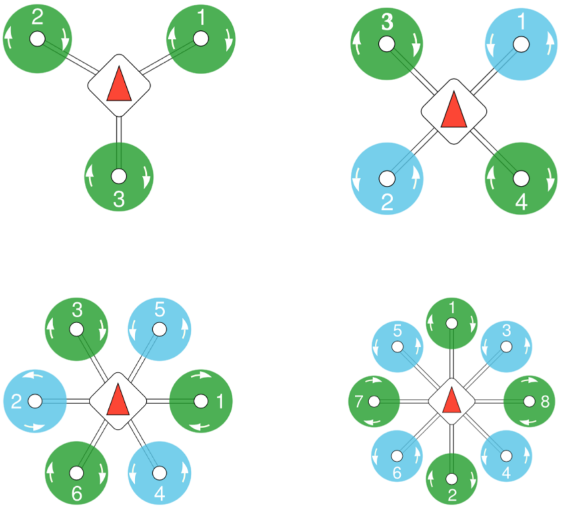


Figure 2.4: Types of multirotors [2]

The control of theses drones is guaranteed by the propellers varying the relative speed of each rotor to change the torque and thrust produced. Like the fixed-wing drones, the multirotors can rotate in the same three axes (Pitch, Roll, Yaw) perpendicular to each other and that intersect the UAV in its center of gravity as shown in figure 2.5.

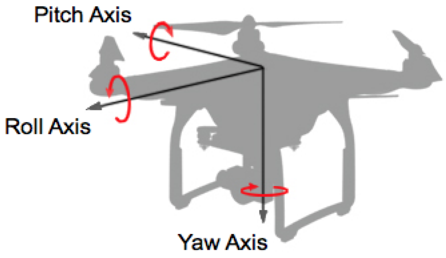


Figure 2.5: Three axes of multirotor drone [3]

One of the most significant advantages of multirotors is their maneuverability compared to fixed-wing drones. This enables them to take off and land vertically requiring less space to take flight. They also have the ability to fly vertically and horizontally, unlike fixed wind drones that can only fly horizontally (forwards), and they can hover at a specified location. Due to the absence of wings, multirotors are generally compact, allowing easier use in confined areas.

Despite all these advantages, the multirotors present a limitation in the flight range on a single battery. Another disadvantage of this type of drones is their vulnerability to the wind, which is countered by the drone itself, spending much more energy than in a windless environment [30].

2.1.1.3 Summary

After analysis and considerations, it is possible to conclude that some of the UGVs have tasks related to human labor in the field helping farmers doing tasks like place seeds, harvest plants, and irrigation and also related to data collection. The UGVs are vehicles with the capacity to carry some weight so they can be equipped with several sensors, actuators, and large batteries. Despite being quite robust, these ground vehicles have limitations concerning movement and accessibility.

In the UAV's universe, most of them have tasks related to data collection using sensors and image processing, but also perform some agricultural tasks. The drones are flexible and can easily reach inaccessible areas. These vehicles are limited by battery life and weight they can carry, being equipped with fewer sensors and smaller batteries.

In the table 2.1 it is possible to see an overview of the main features of the fixed-wing drones and multirotor drones. Considering these two type of drones, the multirotor UAV seems a better option for this project where the ability to remain stable in a specific location and high maneuverability may be an advantage compared to the fixed-wing drones and UGVs.

	Fixed-wing	Multirotor
Maneuverability	Low	High
Portability	Low	High
Range	High	Low
Stability with wind	High	Low
Take off and Land vertically	No	Yes
Hovering in a place	No	Yes
High Speed	High	Low

Table 2.1: Summary table comparing the two type of drones side-by-side

2.2 UAV Platforms

2.2.1 Hardware

2.2.1.1 Customizable UAVs

Drones are a complex fusion of hardware and software. All these systems are combined to form a drone, making it possible to fly and be controlled as desired.

A quadcopter is usually formed by a frame, a flight controller, four electronic speed controllers (ESC), four motors, and a battery. An electronic speed controller is a central processing unit (CPU) to control the speed of an individual motor determining the revolutions per minute (RPM) of each motor in response to the received input. The flight controller is an onboard system that makes it possible to control the drone. A command for the quadcopter to move to the right is received by the flight controller which determinates what motors should be working and their speed. A common flight controller has several integrated sensors like accelerometers, gyroscopes, and barometers to supplement calculations and decisions in order to control the drone. However, only some flight controllers have an autopilot function: software using GPS and position estimator with other sensors allows the drone to fly autonomously to the desired location .

A typical hardware that composes a quadcopter is presented in figure 2.6. In addition to the aforementioned components, some drones can be fully customized according to their purposes. In this customizable drones, several modules like communication module, cameras, external sensors, and others can be added. A module widely used in drones is the visual odometry module. This module is used to estimate the position of a robot, based on an optical flow sensor. In addition, this module can be used to estimate the translation velocity of the drone in relation to the ground, making it possible to keep the drone stable at a specific position, during flight.

Some controllers used in customizable drones that were considered and analyzed for this project will be presented next.

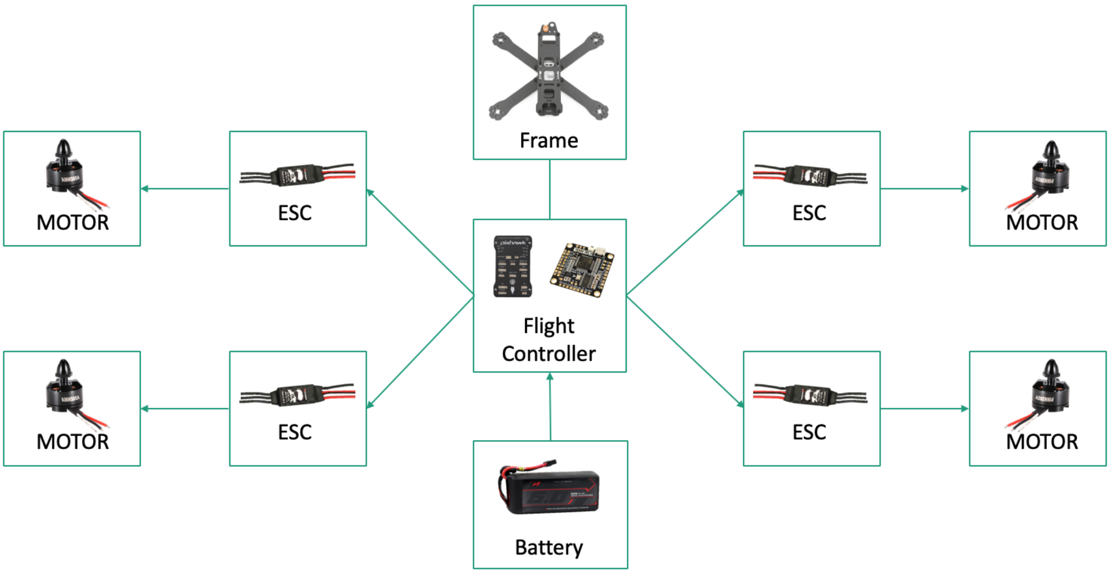


Figure 2.6: Typical hardware of a quadcopter

- **3DR Pixhawk**

The 3DR Pixhawk controller is an open-source flight controller that has excellent flexibility concerning sensors and additional peripherals. This Pixhawk system includes a Unix/Linux programming environment and autopilot functions such as advanced scripting of missions and flight management. It can be configured to fly any airframe from fixed-wing to all types of multirotor.

The system also has a custom driver layer responsible for controlling the inputs and outputs of the peripherals and ensuring secure timing across all processes. This system has abundant connectivity options for additional peripherals, and a micro SD card to expand memory to save data. It also has a built-in backup system for in-flight recovery, redundant power supply inputs, and automatic failover. All these elements make this flight controller safer for outdoor and use in the presence of people [31].

This flight controller is very used for several purposes, and it has a large community that provides support and help in the dissemination of ideas. It supports PX4 and Arducopter that are two open-source firmwares for flight controllers widely used in the drone industry.

- **APM 2.5**

APM 2.5 is a very used flight controller. However, it is at the end of its life. Although this flight controller also has a good support community, it no longer supports the most recent firmware available of PX4 and Arducopter. The last version available of these firmware is v1.9.2 and v3.6.9 respectively. The last version supported by this flight controller is v1.6.5 and v3.2.1 respectively [32]. Considering this, the APM 2.5 was not considered for this project.

- **OpenPilot CC3D**

The OpenPilot CC3D flight controller is a stabilization hardware that can be used in any frame available. It does not support autopilot functions, so it is not possible to use this hardware to perform autonomous flights without adding a specific module for that [33].

This flight controller is usually used in racing drones and drones that only receive commands from the remote controller and executes them. Since autonomous flights are a project requirement, the use of this flight controller was not considered for the development of this project.

2.2.1.2 Ready to Fly UAVs

Currently, selling drones in retail stores is quite usual. The demand for these vehicles by the customer has been increasing. The drones available in this type of market can be called ready to fly (RTF) drones. All the modules used to fly are already connected and configured, so the customer does not need to purchase any additional accessory and to do any configuration or assemble to be able to fly the drone.

Typically, these drones are sold in a package which includes the drone itself and a remote controller, despite some cases where an application installed on a smartphone can control the drone. For this project, several ready to fly drones were reviewed. Each platform has different characteristics that distinguish it from the rest. The examined RTF drones will be presented next.

- **Intel Aero RTF**

The Intel Aero Ready to Fly Drone is a fully assembled quadcopter developed by Intel. It has all the typical hardware of a quadcopter including an Intel Aero Flight Controller with several reconfigurable inputs and outputs providing flexibility to developers and researchers [34]. It presents several visual sensors for visual odometry and avoidance of obstacles, being able to perform a flight in a very stable and safe way.

Once the flight controller is running embedded Linux, it is pre-programmed with PX4 firmware but can be changed to other firmware. The time of flight of Intel Aero can be personalized according to the chosen battery capacity as long as the battery has 3 or 4 cells. Although this drone presents favorable features for this project, it has a high price compared to other RTF drones, costing 985.50 euros.

- **Parrot AR Drone 2.0 Elite Edition**

The Parrot AR Drone 2.0 Elite Edition is a ready to use quadcopter drone that allows the user to remote control it using a mobile application via WiFi. It is typically used to shoot videos and pictures once it has a high-definition camera attached, being incorporated into a low-cost drone market for personal and entertainment use. It has an extensive support community. This drone has a USB port that is used to connect a flash drive to store pictures and videos [35].

All the modules used to fly a drone are already available in AR Drone 2.0 including an odometry module with an altitude ultrasound sensor and facedown camera used to estimate the translation velocity of the drone in relation to the ground. This way, it is possible to keep the drone stable at a specific location [35].

However, this drone does not have a GPS module. There is a similar version that includes a GPS module, but it is no longer sold [36]. It has a small battery (12 min). If it is used outdoor with wind, for example, it will have a shorter duration. It costs 219.00 euros.

- **Parrot Bebop 2**

The Parrot Bebop 2 is a fully assembled quadcopter ready to fly at the time of purchase. It can be controlled using a mobile application or a specific remote controller.

This drone has a large battery (25 min) and a dual-GPS system making the drone very stable and precise even in windy environments. It also has a USB 2.0 port and a serial port that can be used to include additional hardware. It has all the standard modules of a drone, including an integrated odometry module and a WiFi module [36].

This platform carries a flight controller running Linux, becoming possible to add changed firmware in the drone. It is compatible with PX4 and Arducopter firmware.

Bebop 2 has an affordable price comparing to other drones with the same functions, costing 652.25 euros.

- **DJI Mavic Pro**

The DJI Mavic Pro is a compact quadcopter that can be controlled using a remote controller that works in tandem with a smartphone using a proprietary transmission system, called DJI OcuSync. It has a built-in 4K camera, and in addition to all the previously mentioned components, it also has front vision sensors for collision avoidance. Its battery delivers around 27 minutes of flight and can remain stable in windy environments presenting a very smooth and stable flight [37].

Although this drone has many favorable characteristics to its use, all its hardware and firmware can not be changed or adapted to new projects, as it is protected by the manufacturer. It costs 1194.96 euros

2.2.1.3 Summary

All the platforms presented above perform well and could be used in this project.

Considering the full customizable UAVs and taking into account all the features of the presented flight controllers, the 3DR Pixhawk seems to be a good option for this project. Considering the ready to fly drones and weighing the advantages and disadvantages of each one, the Parrot Bebop 2 also seems a reliable option for this project since it presents a balanced price and good performance. In table 2.2 it is possible to see all the examined ready to fly drones compared side-by-side.

The first chosen platform for this project was the 3DR Pixhawk autopilot, which was already available at Fraunhofer AICOS. A process of investigation and adaptation to this platform has taken place over some time. All of the components needed to assemble the drone were validated

	Intel Aero RTF	AR Drone	Bebop 2	Mavic Pro
Reconfigurable Inputs	√	√	√	×
Change Firmware	√	√	√	×
Flight Time	Customizable	≈ 12 min	≈ 25 min	≈ 27 min
GPS	√	×	√	√
Odometry Module	√	√	√	√
Price	985.50	219.00	652.25	1194.96

Table 2.2: Summary table comparing RTF drones side-by-side

using simulations. When passing the tests on the physical platform, undesired behaviors emerged in the control of the drone.

Some time was devoted to solving the problem, and several approaches were taken. Different drone tunning configurations were performed, and all components and their configurations tested. It was found that the problem of drone instability resulted due to the visual odometry module, designed to keep the drone stable in a specific position. The images captured by the module, which uses a face-down camera and an ultrasound sensor to know its location, appeared to be similar to the images posted on the platform's forum used, however, the module did not fulfill its function. After changes in configuration parameters and modifications in the firmware of this module, it was concluded that the problem presented arose from a hardware failure, making it impossible to use.

Since this platform already presented some years, and the odometry module compatible with this controller is already discontinued, new research was carried out. From this research, it was concluded that the best decision would be to acquire a new platform, the Parrot Bebop 2, which, although less versatile, would serve the needs of the project.

2.2.2 Software

To provide a flexible set of tools, the flight controllers have a software running on them implementing hardware support and drone's control.

The most used open-source firmwares in the drone's industry are PX4 from Dronecode and Arducopter from Ardupilot [38].

These two open-source softwares are very similar to each other, having the same functions, and providing support to almost the same UAVs. Despite its similarity, the Arducopter use is not beneficial once all the modifications made on the firmware have to be available to the community. The PX4 firmware allows free use and modifications without the need to reveal the changes.

The use of PaparazziUAV, an open-source firmware with autopilot functions, was also considered. However, this firmware does not have a large support community that can be used to discuss ideas and problems.

Considering the above, the PX4 firmware is the most appropriate software to develop this project being possible to change the firmware without its disclosure and having a large support community available.

2.2.2.1 Ground Station Softwares

To provide access to vehicle setup parameters and other features, a desktop Ground Station Software is often used. A ground station software is typically a software application running on a ground station computer that communicates with the UAV. It displays real-time data of the UAV performance and position. This software can also be used to control the UAV in flight, uploading new mission commands, and setting parameters. It is usually used to monitor video stream from UAV's cameras.

Some of the more popular ground station software compatible with PX4 firmware are Mission Planner and QGroundControl.

- **Mission Planner**

Mission Planner is a ground station application for the Arducopter firmware, but it is also compatible with PX4. This software can be used as a powerful control supplement for autonomous vehicles, and it can load the available firmware into compatible PX4 autopilot boards, and also configure the vehicle. Furthermore, this ground station can plan, save, and load autonomous missions with waypoints entry on Google or other maps [39]. It is compatible with Windows only.

- **QGroundControl**

The QGroundControl software provides full flight control and mission planning for PX4 and Arducopter powered vehicles. This ground station also has mission planning for autonomous flights and flight map display showing vehicle position, flight track, waypoints, and vehicle instruments [40].

QGroundControl runs on Windows, OS X, Linux, iOS, and Android.

2.2.2.2 Summary

The desktop ground stations listed above are very similar and have almost the same functions. The choice was based on the versatility of the operating system where the software can be run. Taking this into account, it is possible to conclude that QGroundControl software is more appropriate for this project.

2.3 Sensoring Platforms in Agriculture

As previously explained, the collected data from probes is very important for the field management and to have better control of the production [41]. Therefore, sensor nodes are placed over the fields to measure some requirements and collect real-time information [42].

Several platforms are available for data acquisition, but only the considered more suitable for this project were evaluated, the IoTiP, Waspmote, and Pycno.

- **IoTiP**

The IoTiP is an embedded system developed by Fraunhofer Portugal and NANIUM S.A for wireless devices. This technology is intended to be integrated into IoT solutions and its array of applications show its robustness and the ability to adapt to several applications [43], such as the use of IoTiP as a detector and classifier of the intensity of physical activity [44] and position-independent activity monitoring solution using IoTiP [45].

This system is composed by a nRF52 Cortex-M4 CPU [46] equipped with Bluetooth Low Energy (BLE) and inertial and environmental measurements units. It also provides a physical interface to enable a modular architecture and add new features such as wireless charging.

This platform also provides a SDK that allows the user to fully customize its firmware for a specific application, a very useful feature for this project.

- **Waspmote**

Waspmote is a Libelium's open-source wireless sensor platform focused on the implementation of low energy consumption in data acquisition. This platform has a modular philosophy, so the user can choose which module is more appropriate. The connection module and the sensor boards should be chosen, having modules for all types of communications and data required.

Smart Agriculture modules allow monitoring of multiple environmental parameters like temperature, humidity, and atmospheric pressure. Since the desired sensors are sold separately from the main hardware, this option becomes more expensive if you want to collect multiple data at the same time.

Theses sensors can be reprogrammed using an open-source SDK and an API. All the collected data is transferred to a Cloud using ethernet, WiFi or 3G interfaces, and also stored in local databases [47].

- **Pycno**

Pycno is a plug-and-play sensor that allows collecting ambient and soil measurements. A set of sensors create a WiFi network with each other, creating a cluster where one of the sensors is responsible to push the data directly to the internet using a 3G/4G module [48]. This sensor can not be changed or modified to accommodate new systems or projects.

- **Summary**

All the analyzed devices are well developed, but not all of them can meet the project's needs. Pycno is a platform that can not be changed for this project and was, therefore, discarded as an option.

Both IoTiP and Waspmote could be used in this dissertation. However, the Waspmote does not have built-in sensors, so it is necessary to buy additional modules for the desired functions, becoming more expensive.

Otherwise, the IoTiP has all the desired communication protocols and sensors for this project, being more favorable its use. Additionally, Fraunhofer AICOS' familiarity with this platform allows further customizability and faster deployment.

2.4 Communication Platforms

For this project, the Bluetooth Low Energy was the communication protocol chosen for the communication between the probe and the drone. Bluetooth Low Energy (BLE) is a low power wireless technology used to communicate between devices. This technology is very used because of its optimized and low power consumption. It is present in all modern mobile devices, and it is possible to find libraries for its use easily [49].

Since none of the considered drones had a Bluetooth interface, it was necessary to choose a BLE module to be integrated into the drone. The Bluetooth module chosen for this project was the ESP32 DevKit which is a low-cost and low-energy module with integrated WiFi and dual-Bluetooth (Bluetooth 4.2 and BLE). It has several GPIOs that can be configured and changed for the solicited project. This module can be programmed using the Arduino IDE programming environment which has libraries available for the BLE use [50].

2.5 Summary

The use of autonomous vehicles in Agriculture is increasing. They are usually used to perform repetitive and tedious tasks like place seeds on the soil, harvest plants, and irrigate the plants, but also to collect data using sensors and image processing. An automated option to collect this data would be the use of probes scattered across agricultural fields storing data and then will be collected by an autonomous vehicle.

Several types of vehicles were considered and analyzed. The UGVs have a high capacity to carry sensors, actuators and batteries. Although these vehicles are robust, they have some limitations in terms of accessibility and mobility.

Considering the objectives of the project, the UAVs seem to be a better option due to their versatility and flexibility. Fixed-wing and multirotor drones were considered. Fixed-wing drones have higher wind resistance and longer range, but have lower maneuverability and can not remain stable at any given location. The multirotors, despite having lower wind resistance and shorter range, have a high degree of handling and can hover in a desired location, being therefore the most suitable for the project.

Considering the customizable drones the 3DR Pixhawk appeared to be a good choice for the project. However, problems occur in the visual odometry module used by this flight controller which prevented its use. It was therefore necessary to choose a new platform.

The final choice was the Parrot Bebop 2 as the platform and the IoTiP as the probe, and also the ESP32 DevKit module for the communication between the previous devices. These three devices are further described in chapter 3.

Chapter 3

System Overview

In this chapter, the approach considered for this project is described, taking into account the system requirements: complete autonomous system, the autonomous flight of the drone, the use of Bluetooth Low Energy to communicate with the probes and also a customizable and versatile system that can be changed and adapted. After a careful analysis of all the possible components for this project and compared with the system requirements, the most suitable ones were selected and are presented in more detail.

3.1 System Components

3.1.1 Parrot Bebop 2

The chosen platform for the proposed solution was the Parrot Bebop 2, that can be seen in figure 3.1.

Figure 3.1: Parrot Bebop 2

This platform is a lightweight and compact video drone, offering 25 min of autonomous flight time and weighing 500g. Bebop's camera films in 1080p full HD using its wide-angle

14-megapixel lens. It has an internal flash memory of 8GB used to store movies and pictures that can be accessed using a micro USB port or a mobile application. It can reach about 16.66 m/s horizontally, 5.83 m/s vertically, and it resists headwinds of up 16.66 m/s. It can fly indoor using visual odometry as position estimator, and outdoor using a fusion of GPS and visual odometry [36].

This quadcopter can be controlled using a mobile application or using its remote controller, both over WiFi. It comes with a proprietary Parrot firmware. Using the SDK developed by Parrot it is possible to create new applications. This SDK does not allow access to the entire firmware or allow profound changes [36].

Once Bebop 2 has a development component, the pre-installed firmware can be deactivated, and it is possible to insert a new open-source firmware and use a hidden serial port to communicate with the drone. With some changes in the open-source firmware, it was possible to configure the serial port to establish communication with the Bluetooth module that was added to the drone.

- **PX4 Firmware**

The PX4 v1.9.2 was the open-source firmware chosen to run on the drone. This firmware has several flight modes providing different levels of vehicle automation and autopilot assistance [4]. Autonomous modes do not require pilot control input, and they are capable of executing automate common tasks like take off and land, but also pre-programmed missions using GPS or accepting input from offboard computer or ground station. The automatic flight modes that the PX4 features are [4] :

- Take off - Vehicle ascends to take off altitude and holds position
- Land - Vehicle lands at the position where the mode was engaged
- Hold - The vehicle hovers at the current GPS position and altitude
- Return - Vehicle ascends to a safe height and then returns to its home position and land
- Mission - Vehicle executes a predefined mission/flight plan that has been uploaded to the flight controller
- Follow Me - Vehicle autonomously follows a user using an Android phone running QGroundControl
- Offboard - Vehicle obeys a position, velocity or attitude setpoints provided over MAVLink by an companion computer

All these flight modes are fully automatic, the remote controller is disabled by default except to change modes. Theses modes use sensors to measure the needed position such as GPS, barometer, and visual odometry.

Manual modes are controlled by the pilot, via remote controller with the assistance of the autopilot. The user inputs are sent directly to the drone where the autopilot automatically stabilizes the attitude and altitude of the drone.

This firmware also has configurable failsafe systems to protect and recover the vehicle in unexpected situations. These systems take into account the battery level, the remote control loss, the vehicle's position, and data link and offboard computer loss. Some switches from the remote control can be configured as safety switches to immediately stop motors or return the vehicle in the event of a problem.

The figure 3.2 provides an overview of the PX4 firmware that is composed of two main layers: the flight stack and the middleware. At the top of the diagram, it is possible to observe the middleware layer while at the bottom the modules of the flight stack layer. The arrows show the information flow between modules.

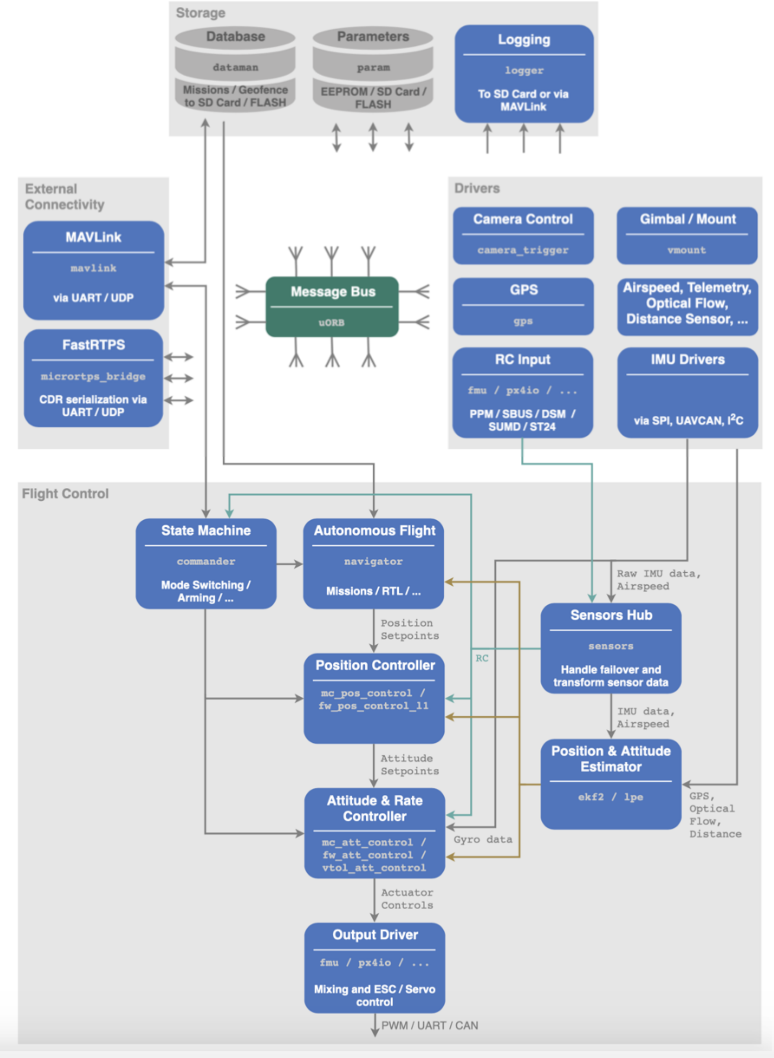


Figure 3.2: Overview Diagram of PX4 [4]

The middleware is a general robotics layer implementing support to any autonomous robot providing external and internal communications, and also hardware support. It consists primarily of device drivers for sensors and communication hardware. Internal communication between the various firmware modules is ensured via uORB. The uORB is an asynchronous publish/subscriber messaging API [4].

The flight stack is a flight control and estimation systems that includes navigation, control, and guidance algorithms for autonomous drones. Figure 3.3 contains the pipeline from sensors, RC inputs, autonomous flight control, to the motor (actuator).

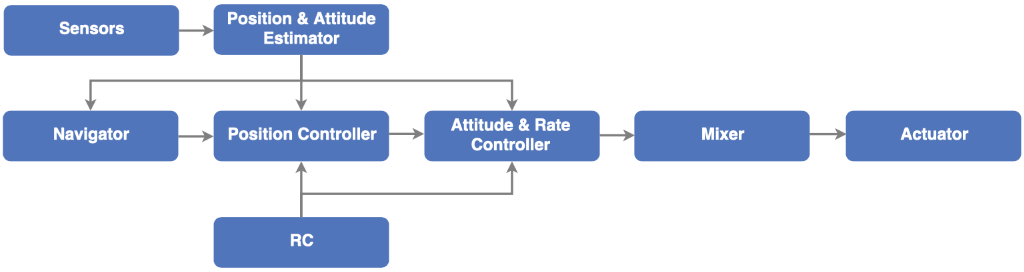


Figure 3.3: Overview Diagram of Flight Stack [4]

The estimator takes sensor inputs and, by combining them, it computes the vehicle state. The controller is a component that takes a setpoint and measurements or estimated states to adjust the value of the process variable to match the setpoint. For example, the position controller receives position setpoints inputs, the process variable is the estimated position, and the output is an attitude and thrust setpoint that move the vehicle to the desired position. The mixer takes commands and translates them into individual motors commands. This translation is different for each type of vehicle and can be influenced by other factors like the center of gravity of the drone [4].

The PX4 flight stack was adapted for this project. It was necessary to create new modules responsible for tasks that would make project realization possible. Some existing modules were adapted so that the new modules were integrated into the firmware. In figure 3.4 it is possible to see the PX4 flight stack with some changes. The two solid green boxes represent the modules that were specifically created for this project. The two light green boxes represent existing modules that have been adapted and changed to receive the new modules.

The Odcdrones module is a module that was created to follow the drone's mission and make decisions based on the geographical location where the drone is located. As soon as the drone is in the desired location, this module sends a request to change the flight mode of the drone, and also sends a command to start the connection with the probe. These commands are received by the Commander module that transforms them into actions along with the Navigator module.

The Serial_BLE module was a module created to take the communication with the Bluetooth module using the serial port, but also to manage the Bluetooth connection with the probes. Through this Bluetooth connection, all the data stored in the probes is transferred to the drone and stored in files.

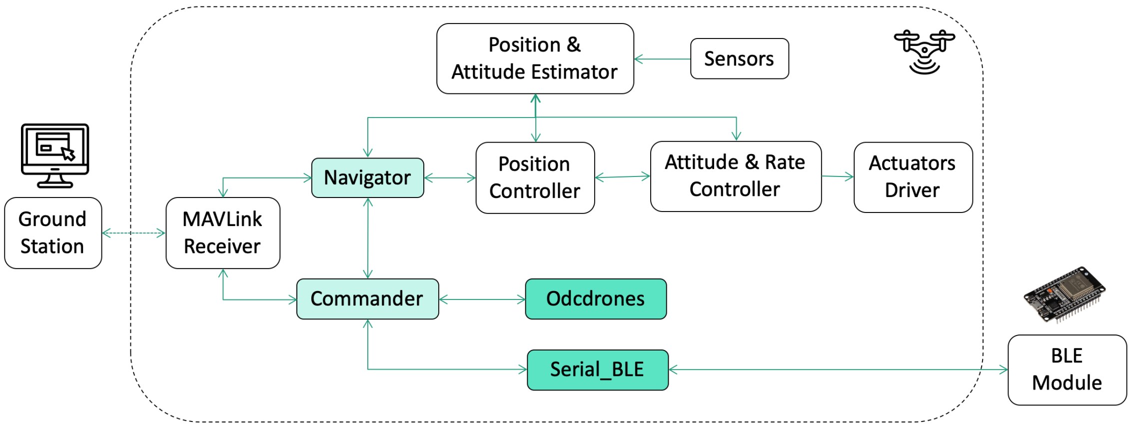


Figure 3.4: Overview Diagram of Flight Stack with new module

3.1.2 ESP32 DevKit

In order to establish a connection between the drone and the probes, it was necessary to choose an efficient and low energy cost communication protocol. The Bluetooth Low Energy (BLE) was the chosen one. Since the Parrot Bebop 2 does not have an integrated Bluetooth module, it was necessary to choose an external module to be adapted to the drone. The external module used was the ESP32 DevKit.

ESP32 DevKit is a development board which is powered by the ESP32-WROOM-32 module. It has a rich peripheral set that includes WiFi and Bluetooth connectivity. It is possible to create rapid prototyping because the ESP32-DevKit has a basic system-requirements already covered. Like a built-in USB-UART bridge, reset and boot-mode buttons, power LED, micro-USB connector, and a set of GPIOs available to the developer, as can be seen in 3.5 [51].

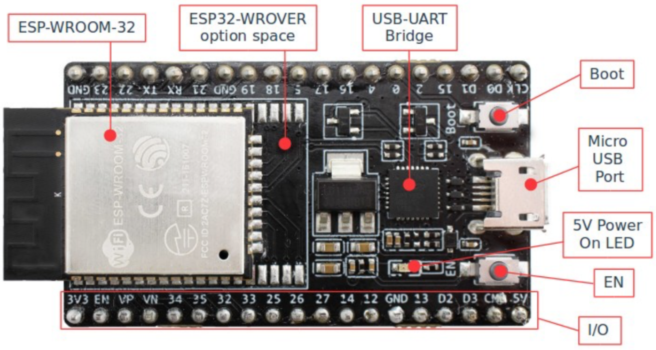


Figure 3.5: ESP32 DevKit [5]

This board was programmed using the Arduino IDE programming environment which has libraries and online support available for BLE use.

A program was created, taking into account the needs and objectives of the project. This program is composed of two parts, each related to a type of communication. There is a section where the serial port is configured to be able to communicate with the drone. In addition to the port configuration, this part is responsible for receiving the drone commands. The other part was created to turn the commands into actions related to Bluetooth communication with the probe, considering the IoTiP firmware architecture.

3.1.3 IoTiP

Internet of things in a Package (IoTiP), shown in figure 3.6, is a development platform that integrates software, firmware and hardware components to produce a development ecosystem for the IoT, created by Fraunhofer Portugal AICOS and NANIUM S.A. [43].

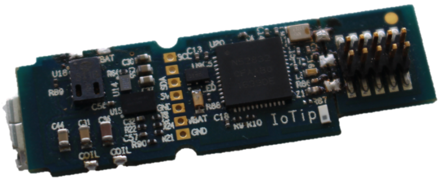


Figure 3.6: IoTiP

This platform is small in size, low in energy consumption, good in integration with other systems, and low in costs. It features environmental measurements units, being able to measure temperature, pressure, relative humidity, and air quality. These measurements can be expanded by adding external sensors to IoTiP.

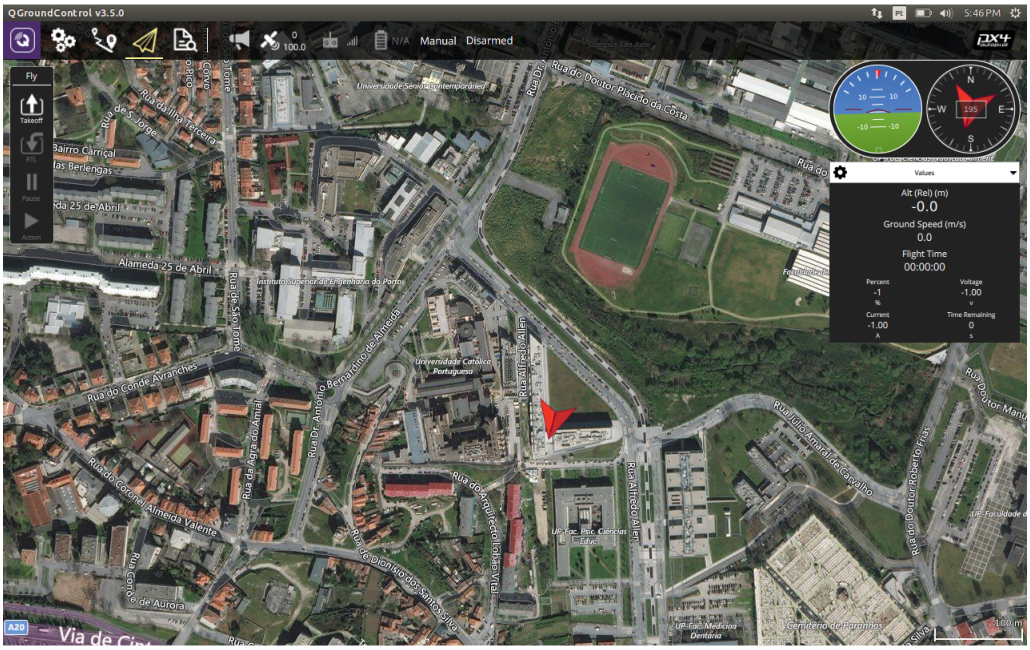
The IoTiP ecosystem embeds sensing, processing, energy management, and communications. It also provides a physical interface that permits a modular architecture, being possible to add new features for new applications and a hardware abstraction layer to reduce the development cycle [43].

It is composed of a Nordic Semiconductor nRF52832 that supports Bluetooth 5. It has an ARM Cortex M4 CPU, running at 64 MHz being capable of dealing with communication tasks in a short timeframe. This microprocessor features a flash memory with a capacity of 512kB and a 64kB RAM memory [46]. Part of this memory is used for storing the firmware, and the remaining can be used to store sensor data, being possible to store a maximum of 7500 samples of the relative humidity sensor. If the samples were stored thirty minutes apart, about five months of data can be stored. In this project, only the relative humidity sensor was used as proof of concept.

The firmware performs specific actions taking into account the commands received by the device that has connected to it. There is a command for starting the data transfer, and another command to set it in its normal mode of operation.

3.1.4 Ground Station Control

QGroundControl provides mission planning and flight control for MAVLink enabled drones, shown in figure 3.7. MAVLink is a messaging communication protocol very used to communicate with drones. It follows a publish-subscribe and point-to-point pattern: data is published as topics while mission protocols and parameter control are transmitted point-to-point [52]. The PX4 firmware uses the MAVLink protocol to communicate with the ground station software, over WiFi.



Alt (Rel) (m) -0.0
Ground Speed (m/s) 0.0
Flight Time 00:00:00
Percent -1 %
Voltage -1.00 v
Current -1.00 A
Time Remaining 0 s

Figure 3.7: QGroundControl

With QGroundControl, it is possible to access a set of configurations of PX4 powered vehicles and see a flight map display showing real-time vehicle position, waypoints, and vehicle instruments. Furthermore, this software is used to upload the firmware to the drone, tune the vehicle, and create autonomous missions [51]. With its use, the user can change the flight mode of the drone, see in real-time the status and measurements of the sensors present in the vehicle, and specify safety behaviors.

Besides the use of QGroundControl, a script was developed to download all data files from the drone to the computer. Each of these files contains the stored data of each probe and probe's information then converted into a plot for more accessible analysis.

3.2 Components Integration

In order to form a functional system using the components presented, it was necessary to integrate the components using different protocols. In figure 3.8, it is possible to see a schematic of the communication protocols used between each component.

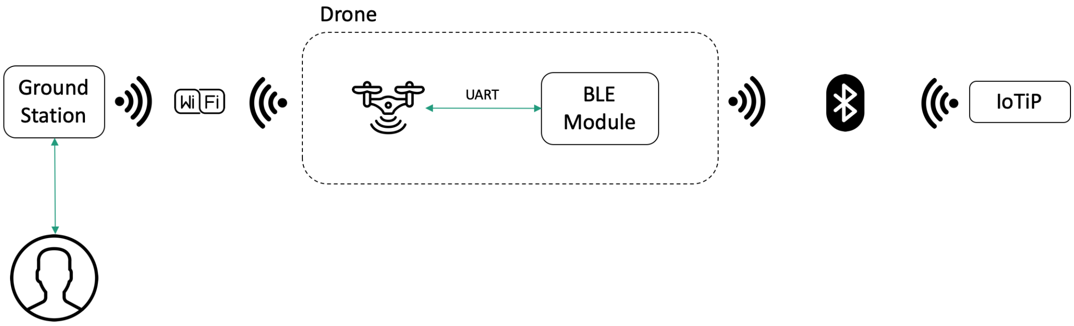


Figure 3.8: Overview system communication protocols

The ground station software is running in the user's computer, and it communicates with the drone via WiFi. This communication allows real-time data exchange of vehicle behavior and pilot commands. Between the drone and the external Bluetooth module that has been added for this project, communication is done using a UART serial port, as shown in figure 3.9. This way the on-board system, composed by the drone and the Bluetooth module, was created.The communication between the vehicle and the IoTiP was assured using Bluetooth Low Energy.

Figure 3.9: ESP32 DevKit integrated into Bebop 2

In this way, all the components can communicate with each other creating a functional system where the data can efficiently flow.

3.3 Summary

The system for this project was built using hardware that was adapted to fit its needs. Each component presented in this chapter came from analysis and choice based on the state of the art.

Bebop 2 was the leading and most important choice for the project. It is a very stable drone, which features all the components needed to fly outdoors. It can be uploaded with the PX4 open-source firmware, increasing the flexibility and expansion of its use. This drone also has a serial port which, when configured through the firmware, increases the possibility of integrating the drone with other modules such as the ESP32 DevKit module.

This module appears in the system to establish communication using Bluetooth Low Energy between the drone and the probes. It can be programmed using the Arduino IDE programming environment allowing easy and fast customization.

IoTiP was used as a probe, since it presents a low energy consumption, offers a set of sensors that can be used for environmental measurements, and uses Bluetooth Low Energy to communicate. To be able to monitor the state of the system in real-time and to control the drone, the QGroundControl was used as ground station control interface.

Chapter 4

Autonomous Opportunistic Data Collection

4.1 Onboard System

In order to develop a system that performs an opportunistic data collection, as described in the chapter 1, special consideration took place in order to develop the system onboard. All the processing, analysis, storage, and management of connections was developed to be performed by the drone. Thus the system becomes more autonomous and independent since it does not need any offboard unit for the execution of the mission.

4.1.1 Drone

As mentioned in section 3.1.1, the drone performs the planned functions as a result of the changes made on its firmware.

The PX4, as described earlier, features several autonomous flight modes. The Mission Flight Mode allows the drone to execute a previously defined flight plan composed of geographic coordinates where the drone has to pass. Although this flight mode includes the hypothesis of the drone performing the Hover maneuver in each of the waypoints, the maneuver time must be set before the start of the mission.

In the case of this project, the desired behaviour is that the drone performs the Hover maneuver, indefinitely since the transmission time between the Bluetooth module and the probe can be variable, depending on the amount of data stored in the probe. If the Bluetooth module can not connect to any probe, the maneuver stops.

During this time, it is necessary to perform tasks on the Bluetooth module. After all these tasks are performed, and all probe data is present in the drone, the Hover maneuver can be terminated, given the continuity of the previously executed mission.

Taking into account this essential factor for the operation of the system, it was necessary to adapt the firmware so that it could autonomously and in real-time change the mission that was previously predefined. No firmware function allowed the real-time change of the mission, without the intervention of the pilot or an offboard system that sent a new mission altered, so it was necessary to create a method that fills this project need. In figure 4.1 is presented the state diagram of the drone's firmware. The state in green is a state that was developed to change the state of the drone and to manage the connection with the IoTiP over Bluetooth.

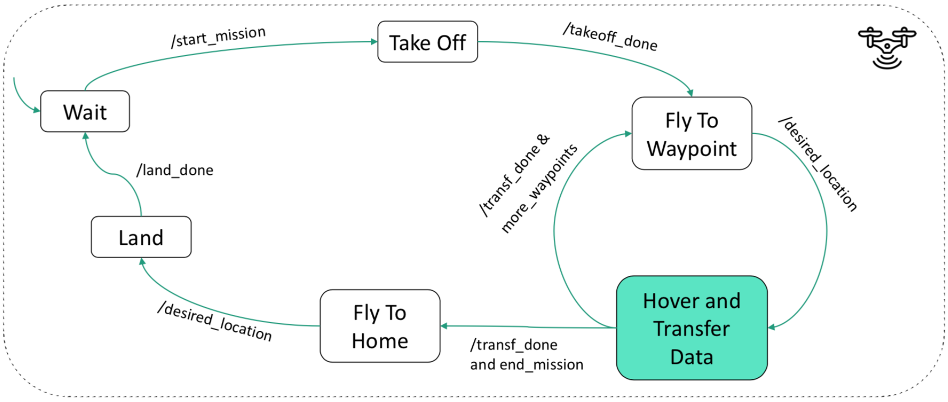


Figure 4.1: State Diagram of Drone's firmware

Another important task for the project was the need to integrate the Bluetooth module into the drone. The drone features a micro USB port, which is not being used for any function, and an internal serial port which also does not play any function in the firmware. Being the serial port a port of more straightforward configuration, thus speeding up the process of integration of the Bluetooth module. In terms of hardware, it was necessary to physically integrate the Bluetooth module into the drone, using the serial port in the drone for communication with the probe and also the micro USB port for powering the Bluetooth module.

As explained in section 3.1.1 using figure 3.4, new modules have been added and integrated into the remaining firmware. All changes will now be explained in more detail.

The Navigator module is a module that, in cooperation with the MAVLink receiver, receives the mission that is sent by the ground station control (QGroundControl). The mission is validated by the Navigator to check if all mission commands are possible to execute and if safety distances are met. In case the mission is not valid, the flight controller discards this, and the pilot receives warnings according to the errors present in the mission. In case the mission is valid, the Navigator sends to the Commander the GPS coordinates of the waypoint, which are used fly to that point.

The Commander is a module responsible for controlling the drone's flight mode. When it receives valid coordinates of a mission, it places the drone in Mission Flight Mode by starting the mission.

These two modules had to be adapted when the new Odcdrones and Serial_BLE modules were created. Also, new uORB topics had to be created for communication between the new modules. The new uORB topic can be created in the main firmware repository. All commands that are sent between modules use specific uORB topics. The module that wants to send a command publishes the desired variables and commands in the topic. The module that needs that information has a subscribed topic and performs a particular action when the topic is changed.

The new Odcdrones module has been created to check if it is necessary to put the drone in Hover mode or not. This module checks through the Commander if the drone is in mission mode. If this option is real, the module will continuously check if the drone is in the desired waypoint or not. After checking the location of the drone, and if its position is the desired, Odcdrones sends a command to the Commander to change the flight mode from the Mission Mode to the Hold Mode. In this mode the drone remains stable in the current position and at the same height, performing the Hover maneuver.

Once the flight mode has been changed to Hold Mode, the Commander sends a command to the Serial_BLE module to activate the Bluetooth module and perform its functions.

After that, the Commander waits for the Serial_BLE module to send a command confirming the end of the IoTiP data transmission and to allow the mission to continue. Once this command is received, the drone's flight mode is changed back to Mission Mode, causing the drone to advance to the next waypoint. This process is repeated until the end of the mission when the drone returns to its starting place and lands.

The Serial_BLE module was created to communicate with the Bluetooth module using the serial port and also to receive and store all data sent by IoTiP. When the drone is powered on, this module starts automatically and activates the serial port, with the desired settings, which are in the table 4.1. From this moment, this module is operational and ready to receive commands from the Commander.

Baud Rate	115200
Parity	None
Data Bits	8
Stopbits	1
Flow Control	None

Table 4.1: Serial port setting

When the drone is at the probe's location and in Hold Mode, the Commander sends a command to initiate the Bluetooth functions. This module sends several commands, via serial port to the Bluetooth module and whenever a command is sent, the Serial_BLE module awaits confirmation by the Bluetooth module. The first command sent contains the MAC address of the probe located at that point that is used by the Bluetooth module to search the probe and establish a connection with it. When the confirmation and information is received, a new command is sent to the Bluetooth module until all tasks are performed. This whole process of communication between the modules is represented in figure 4.2.

All information received by the Bluetooth module is sent to the Serial_BLE module where it is stored in files. For each probe, a different file is created, and at the end of the mission, there are as many files as the number of probes visited.

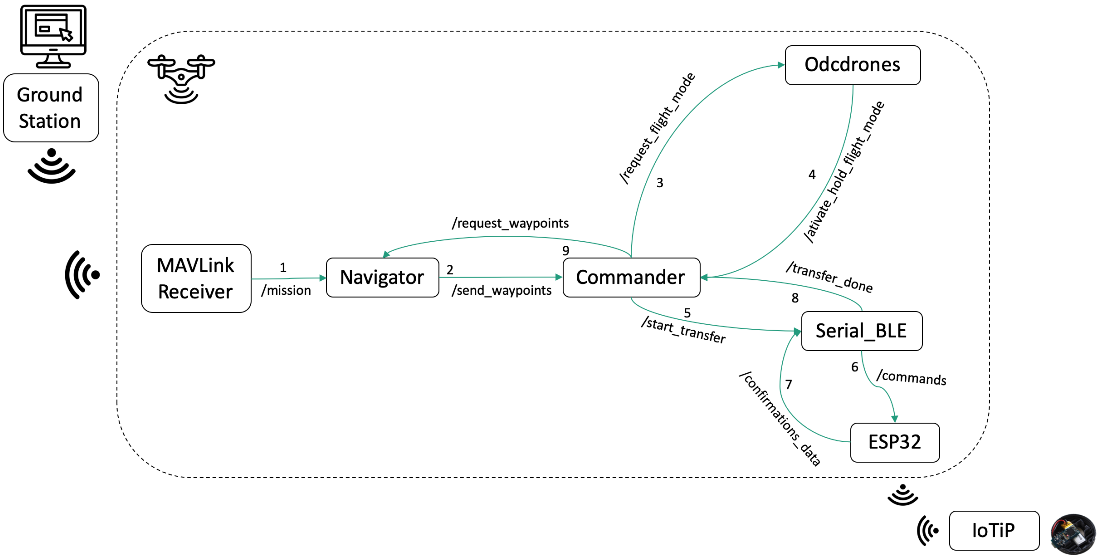


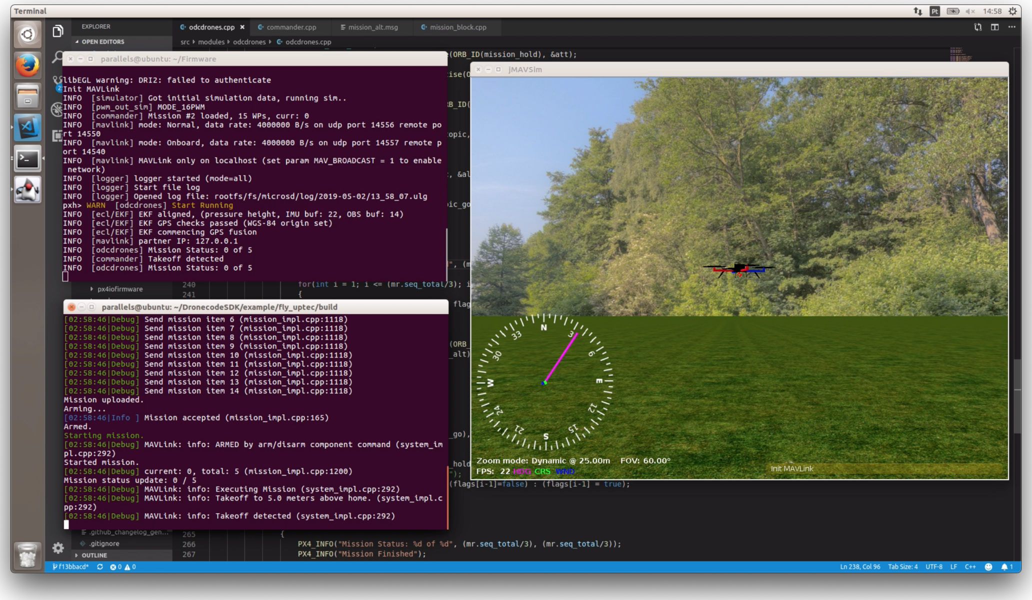
Figure 4.2: Scheme of communication between modules

The main data that is stored in the file is the waypoint number, accompanied by the name, MAC address, and battery level of the probe. The average signal power of the probe is also saved at the time of data transfer.

While awaiting the arrival of the new platform, these modules were developed and tested in simulators. The PX4 provides a simulation version of the flight controller and the vehicle. The vehicle simulation was made using the JMAVSim simulator [53], that already comes with all the necessary configurations to simulate a quadcopter. In this simulation, the command to continue the mission was inserted manually, since the main objective was to test the flight. The figure 4.3 shows the simulation environment used.

When the new platform was already available in Fraunhofer, it was necessary to configure it and then upload the changed firmware. Several changes were required in the modes of operation of the modules created since the firmware version used for the simulations was older than the version supported by Bebop 2. This new version does not allow simulation and presents a different firmware organization, being necessary a new study in order to be able to adapt and integrate the previously created modules.

After all the necessary changes, the first experimental drone flight with the custom firmware was performed. In this flight, the drone presented a behavior slightly different from the expected one, since that, in the simulation, not all the aspects are simulated. The size and weight of the drone, as well as the presence of wind and small irregularities, are examples of aspects the simulator does not considered. In addition to all the firmware changes, parameters have also been changed to ensure failsafe actions. As an example, the drone performs the land maneuver if the battery level drops below 10%, and it also performs the same maneuver if the communication with the ground station is lost. It was also configured an external remote controller connected to the ground station with an emergency button that turns off all drone motors as soon as it is pressed.



```
libEGL warning: DRI2: failed to authenticate
Init MAVLink
INFO  [simulator] Got initial simulation data, running sim..
INFO  [pwm_out_sim] MODE_16PWM
INFO  [commander] Mission #2 loaded, 15 WPs, curr: 0
INFO  [mavlink] mode: Normal, data rate: 4000000 B/s on udp port 14556 remote po
rt 14550
INFO  [mavlink] mode: Onboard, data rate: 4000000 B/s on udp port 14557 remote p
ort 14540
INFO  [mavlink] MAVLink only on localhost (set param MAV_BROADCAST = 1 to enable
 network)
INFO  [logger] logger started (mode=all)
INFO  [logger] Start file log
INFO  [logger] Opened log file: rootfs/fs/microsd/log/2019-05-02/13_58_07.ulg
pxh> WARN  [odcdrones] Start Running
INFO  [ecl/EKF] EKF aligned, (pressure height, IMU buf: 22, OBS buf: 14)
INFO  [ecl/EKF] EKF GPS checks passed (WGS-84 origin set)
INFO  [ecl/EKF] EKF commencing GPS fusion
INFO  [mavlink] partner IP: 127.0.0.1
INFO  [odcdrones] Mission Status: 0 of 5
INFO  [commander] Takeoff detected
INFO  [odcdrones] Mission Status: 0 of 5
```

```
[02:58:46|Debug] Send mission item 6 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 7 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 8 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 9 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 10 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 11 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 12 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 13 (mission_impl.cpp:1118)
[02:58:46|Debug] Send mission item 14 (mission_impl.cpp:1118)
Mission uploaded.
Arming...
[02:58:46|Info ] Mission accepted (mission_impl.cpp:165)
Armed.
Starting mission.
[02:58:46|Debug] MAVLink: info: ARMED by arm/disarm component command (system_im
pl.cpp:292)
Started mission.
[02:58:46|Debug] current: 0, total: 5 (mission_impl.cpp:1200)
Mission status update: 0 / 5
[02:58:46|Debug] MAVLink: info: Executing Mission (system_impl.cpp:292)
[02:58:46|Debug] MAVLink: info: Takeoff to 5.0 meters above home. (system_impl.c
pp:292)
[02:58:46|Debug] MAVLink: info: Takeoff detected (system_impl.cpp:292)
```

```
for(int i = 1; i <= (mr.seq_total/3); i
{
PX4_INFO("Mission Status: %d of %d", (mr.seq_total/3), (mr.seq_total/3));
PX4_INFO("Mission Finished");
```

Figure 4.3: Simulation environment.

Several changes were made to the firmware, so the drone performs a steady, precise, and safe flight like:

- Creation of a new state in drone firmware
  - Creation of the new Odcdrones module.
  - Creation of the new Serial_BLE module.
  - Adapting the remaining firmware to the new modules.
- Low battery safety parameter setting.
- Emergency button configuration for stopping drone motors.
- Creation of safety commands that allow the user to move the drone to the next probe.

4.1.2 BLE Module

As mentioned earlier, it was necessary to add a Bluetooth module to the drone to establish communication with the probes. The chosen module, the ESP32 DevKit, is presented as a platform for development and is therefore programmable using the Arduino IDE. We use the ESP32_BLE_Arduino library that are available on the web for use in this module. These libraries provide for the implementation of Bluetooth Low Energy support for the ESP32.

The most commonly used functions in BLE are already implemented, such as device searching, connection, and search for services and characteristics of remote devices. Many of these functions were used for this project, some of which needed some adaptation. All communication between the functions used, management of the received commands, configuration and management of the serial port were developed according to the project.

Once the ESP32 is powered by the drone when it is turned on, this module also switches on automatically. The first task that is performed by the module is the configuration of two pins to function as a serial port. These two pins are connected to the serial port of the drone, so they need to be with the same settings. From this moment, the module is continuously waiting for commands to perform specific actions.

The firmware architecture implemented in ESP32 is represented in the figure 4.4. As it is possible to observe, this module performs several functions after receiving commands from the drone, more appropriately by the Serial_BLE module described before.

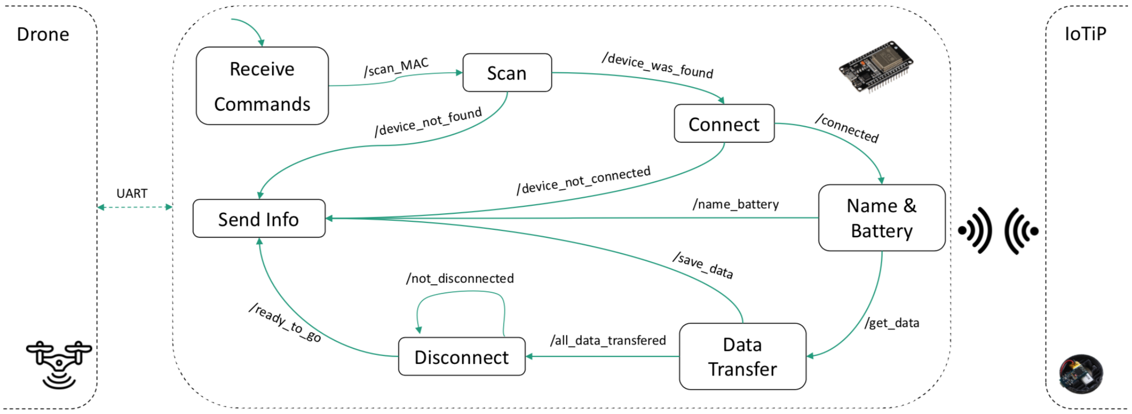


Figure 4.4: ESP32 firmware architecture

The command received by the ESP32 is a command to start the device search. This command includes the MAC address of the probe present in that location, thus restricting the search. This association between the probe's MAC address and its location is done manually in the firmware in the Serial_BLE module. Once the search is done, and the device is found, this information is sent to the drone. The module is now ready to make a connection to the desired device, requiring a confirmation from the drone to establish the communication. After that confirmation, the communication is established, and the drone sends a command asking for the name and battery level of the probe. The module, which is currently connected to the probe, looks for the services and characteristics where this information is available, using the functions of the library. After this data is obtained, they are sent to the drone.

Next, the drone receives this information and is ready to receive the data that is stored in IoTiP. The ESP32 module receives a drone command to begin the data transfer from the probe. At this point, the Bluetooth module changes the value of a pre-selected characteristic of the IoTiP to request the data. Changing the value of this feature indicates that the Bluetooth module is ready to receive the data.

Each time the value of the characteristic is changed, the IoTiP sends a stored value to the Bluetooth module. After the received value is confirmed and sent through the serious port to the drone to be written to a file, the ESP32 requests the next value, thus ensuring that all the values stored in the IoTiP are received. This process is repeated until IoTiP no longer has stored values to send. In the end, the value of the characteristic is restored by the Bluetooth module indicating to the IoTiP that it can start the storage of new data.

Once received the data one by one, it is sent through the serial port to the drone. When the drone already has all received data stored in a file, it sends a command to the Bluetooth module so that it disconnects from the remote device.

This module has been configured to serve as an intervener between the drone and the probe. So it does not store any information about the IoTiP. It only receives commands coming from the drone, performs tasks, and sends the received data to the drone.

All data management described in the previous paragraphs has been developed for this project to ensure the transfer of all its data and correct storage. In figure 4.5 it is possible to see the sequence diagram of the data transfer between drone and IoTiP, explained above.

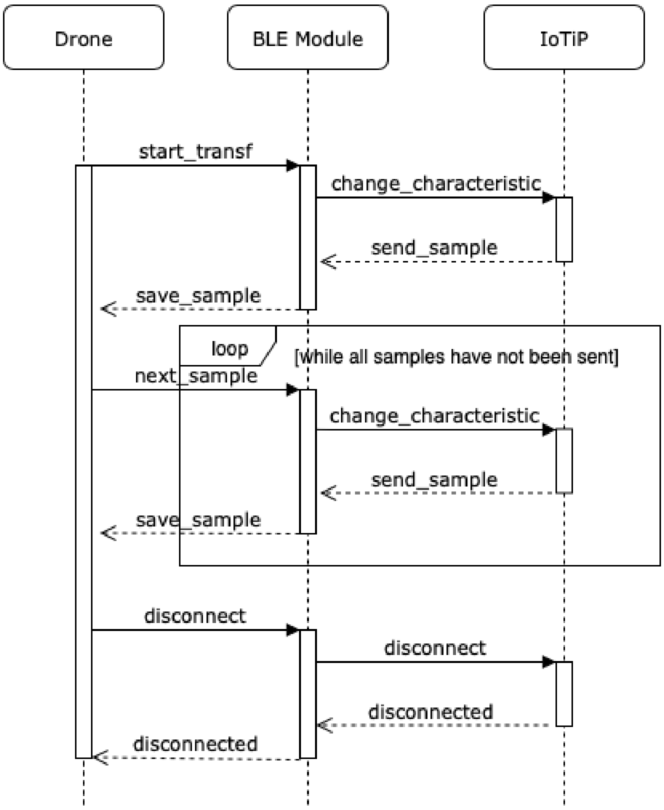


Figure 4.5: Data Transfer Sequence Diagram

4.2 Probe

The probe used for this project was the IoTiP developed by Fraunhofer. As stated in section 3.1.2, this device is quite versatile and can be configured and programmed to meet the desired needs. The IoTiP firmware can be changed and adapted using the IoTiP SDK. This platform allows access to all firmware files.

For this project, some changes were made, so that the IoTiP performed the desired tasks. To change the firmware, it was necessary a learning and adaptation process to the IoTiP SDK, so that the changes made did not affect its performance.

The IoTiP features several sensors that could be used in this project, but using only the relative humidity sensor was considered enough to serve as proof of concept of the system. To store as much sensor data as possible, an array to store 32bit samples, with a maximum capacity of 7500 was created, acting as a buffer.

This sensor has a sampling frequency from 40 Hz to 0.1 Hz.

Considering the frequency of storage equals the sampling frequency, in the case of the maximum frequency (40 Hz), it is possible to store data related to approximately 3 minutes. If the minimum sampling frequency (0.1Hz) is used, IoTiP stores data of approximately 20 hours. However, the storage frequency may be less than the sampling frequency of the sensor, being possible to increase the time between stored data. For example, if the data it stored one minute apart (frequency of 0.016 Hz), it is possible to store data of approximately 5 days.

Figure 4.6 illustrates the IoTiP firmware architecture. It is possible to observe that as soon as the IoTiP is powered, it starts to store data of the relative humidity sensor, taking into account the chosen frequency of storage. Since then, the buffer is being filled with sensor values.

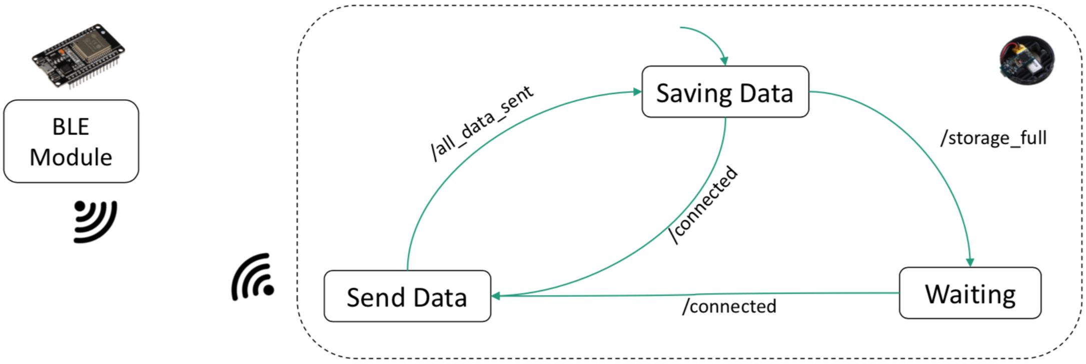


Figure 4.6: IoTiP firmware architecture

When the drone connects to the probe, it changes the value of a preset characteristic. Then the IoTiP automatically stops the store process and sends all the data stored to the drone. After the data has been sent, the buffer is cleared.

When the drone receives all the stored data, it changes the value of the characteristic again, indicating the IoTiP that it can restart the storage.

If the buffer reaches its maximum capacity before the drone collects the stored data, storage is stopped until the data is collected by the drone. After the collection, all data is erased. If the drone connects to the probe without it having its maximum capacity reached, the storage process is stopped, and all data stored so far is sent and subsequently erased from IoTiP. The method described above was developed for this project in order to manage the storage of IoTiP and not to delete samples from the buffer.

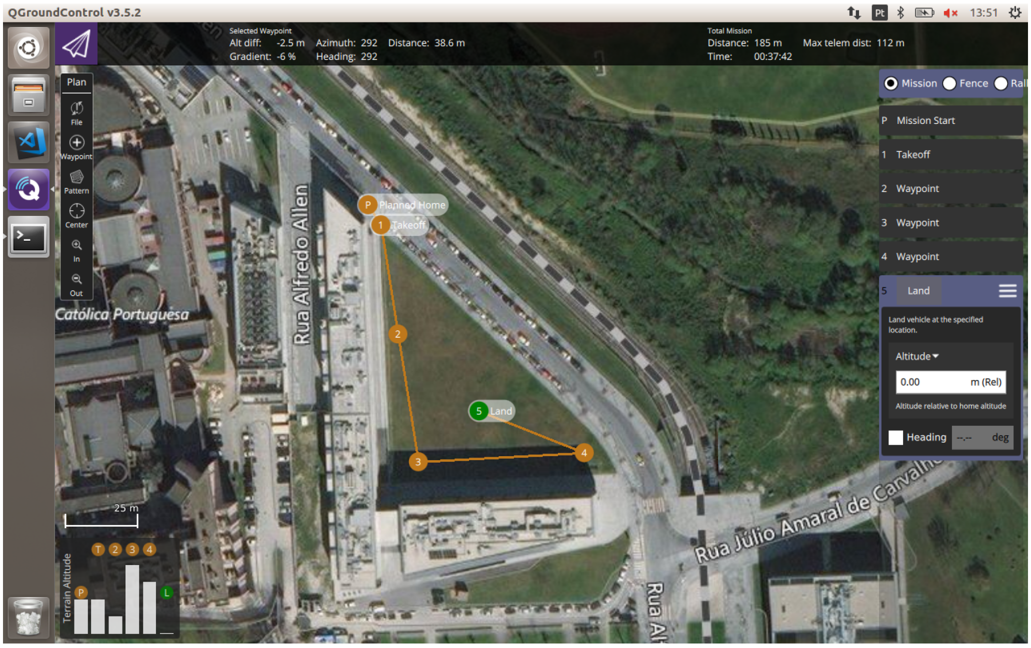
4.3 Ground Station

QGroundControl has been used as a ground station so that the user has access to vehicle status in real-time. This software was not changed for this project. It was only used with functions that were already available.

In QGroundControl the user has access to all parameters that can be changed without having to explore the firmware. This software presents a section dedicated to the flight plan creation that is then sent to the drone.

For the creation of flight plans, the user has to indicate the place where he wants to start the mission, in this place will be the waypoint for the take off maneuver. For this maneuver, it is necessary to enter the desired altitude and geographic location. The next waypoints of the flight plan are defined by geographical coordinates, the relative altitude to the ground and flight speed. The order of insertion of the waypoints corresponds to the order of execution of the mission. Once all desired waypoints are inserted, the user places the final waypoint, which can be of two types. A waypoint to perform the landing maneuver, where the drone lands at that geographical coordinates. Alternatively, a waypoint to send the vehicle back to its starting position and then landing at that location. When the flight plan is complete, it is possible to see the total distance of the plan and also an estimation of the required time to complete the mission, taking into account the chosen speed of each waypoint.

The figure 4.7 shows an example of a flight plan created in QGroundControl.



QGroundControl v3.5.2

Selected Waypoint
Alt diff: -2.5 m
Azimuth: 292
Distance: 38.6 m
Gradient: -6 %
Heading: 292

Total Mission
Distance: 185 m
Max telem dist: 112 m
Time: 00:37:42

● Mission ○ Fence ○ Rall

P Mission Start
1 Takeoff
2 Waypoint
3 Waypoint
4 Waypoint
5 Land

Land vehicle at the specified location.

Altitude
0.00 m (Rel)
Altitude relative to home altitude

☐ Heading --.-- deg

Figure 4.7: Flight plan example

On the computer running this software, there is a script that was created, and it uses an ADB shell that transfers the files containing the probe's data. Through this method, it was possible to create a routine that, when executed, transfers all the desired drone files to a predefined folder on the computer. After this transfer is completed, a Python program is automatically executed to transform the data displayed in the files into plots. The user is asked which probe wants to view, and the plot is displayed and then saved on the computer.

This way, the user can see all the displayed values and their evolution faster and easier. The figure 4.8 is an example of a plot showing the stored data of a probe. It is possible to see the MAC address, name, battery level, and sensor stored values of the probe.

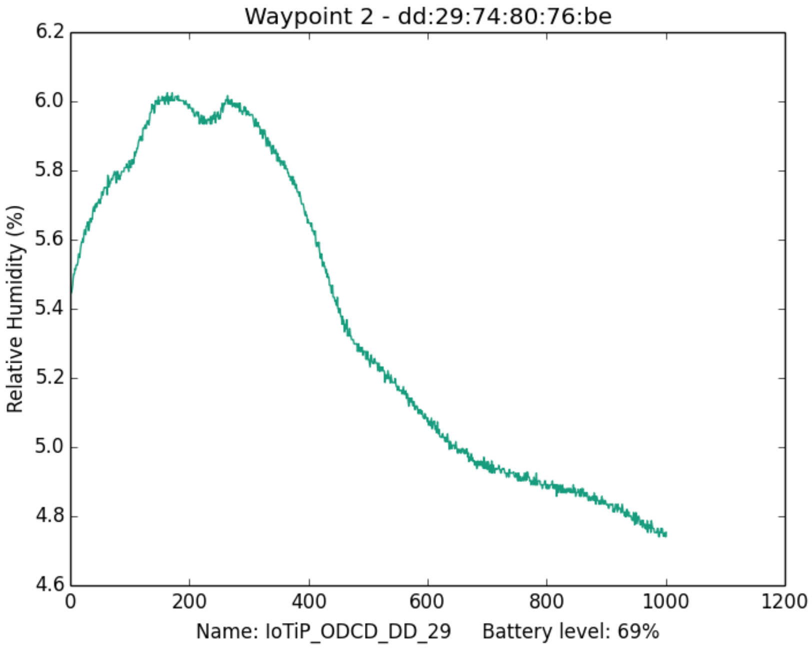


Figure 4.8: Plot of stored data of a probe

Besides using QGroundControl, it is also possible to access the drone using WiFi Telnet to receive more detailed data of real-time firmware behavior.

4.4 Summary

After familiarization with the firmware chosen for Bebop 2, it was necessary to perform several changes to achieve the system objectives. It was necessary to create two new modules and integrate them into the remaining firmware.

A module has been created to follow the mission and at the desired waypoints, change the flight mode of the vehicle to Hold Mode to make the drone stable in the desired location. In addition to this module, another one responsible for serial port communication with ESP32 was also created.

The remaining firmware modules have been adapted to communicate with the new modules.

Regarding the Bluetooth module that was added to the drone, a firmware was created that perform actions and communicate with IoTiP using Bluetooth Low Energy. From this communication comes information and data of the probes, which are stored in files in the drone. The IoTiP firmware also had to be adapted to have a buffer that stored the relative humidity sensor data. Whenever a connection is made by the drone, the contents of this buffer are sent to the drone and then erased from IoTiP's memory.

At the Ground Station, QGroundControl is used to monitor the vehicle condition, create flight plans as desired, and also configure the drone. In addition to QGroundControl, a script was created to transfer the drone files to the user's computer using the ADB shell. Besides transferring files, this script is also responsible for displaying the stored data in plots, using Python.

Chapter 5

System evaluation and discussion

5.1 Drone Behavior

In order to test system performance, a flight plan was created (figure 5.1). This flight plan consisted of two waypoints, where probes were located. The mission was started by sending the drone flight plan through QGroundControl. Once received, the drone started the mission, beginning by moving to the first waypoint. There, the drone remained stable to create a connection with the probe. All data stored in the probe was collected by the drone and, at the end of the transfer, the drone resumed the mission and navigated to the second waypoint. The same procedure was performed to collect data from the probe present at this location. At the end of the transfer, the drone returned to its initial location. Several flights were made to prove the constant behavior of the drone. One of the flights was recorded and the video can be seen at the following link `https://youtu.be/7QiuxLZbE4c`.

Whenever the drone performs a flight, whether manual or autonomous, a log file is created, that records detailed vehicle status and sensor data. These files can be analyzed after the flight to check for existing problems and unexpected behaviors.

Throughout the development of the project, these logs were kept and analyzed, checking if the firmware was working as expected and helping to solve some problems. These files can be extracted from the flight controller using QGroundControl and further analyzed using an online tool called Flight Review [54]. This tool uses the data stored in the file to graph the evolution of the drone behavior. A map is also presented with the drone flight and the proximity of the points where the drone passed compared with the desired ones.

Some of the graphics presented by this tool refer to Drone Performance. In these plots, the red line corresponds to the estimation and the green line to the setpoint. These two lines should closely match if the applied settings are appropriate.

In figure 5.2 it is possible to observe the trajectory that the drone performed in one of the test flights. Also in figure 5.3 a small scheme is shown presenting the estimated position of the drone, the desired, and the position projected by the GPS. In this case, the three lines coincide for much of the path, revealing that the settings applied to the drone are appropriate. In appendix A are

Selected Waypoint
Alt diff: 4.0 m Azimuth: 153 Distance: 3.0 m
Gradient: 93 % Heading: 131

Total Mission
Distance: 115 m Max telem dist: 31 m
Time: 00:37:09

Take off from the ground and travel towards the specified position.

Altitude
4.00 m (Rel)
Altitude relative to home altitude

☐ Heading --.-- deg

Figure 5.1: Flight plan tested

presented the axis (Roll, Pitch, Yaw, x, y, z) and velocity plots created by Flight Review of a log file of a test flight.

Figure 5.2: Trajectory performed by the drone

5.2 Bluetooth transfer

The efficiency of data transmission between the drone and the probe influences the behavior of the entire system. The data transmission time determines how long the drone has to remain stable at each waypoint in order to collect all data.

If the drone has to remain a long time in the Hover maneuver on each probe, the mission execution time will be extended. The longer the drone has to keep in flight, the more battery it will spend, and it may not have enough battery to carry out the entire flight plan or to return to the starting position.

If the IoTiP is fully buffered, 7500 samples of the relative humidity sensor are stored. Transmitting all this data one by one as explained above takes about 3.75 minutes, this value was taken from a drone log recorded on an experimental flight. This method of data transmission is time-consuming, since for each value transmitted from the IoTiP to the drone a confirmation of the value is made, and only then the next value is sent.

Admitting a flight plan with three probes, the total mission time will be increased by 11.25 minutes, considering all probes with full storage and an environment without interference from external factors that may disturb the transmission. This data transmission time limits the number of probes that can be visited by the drone on a flight.

5.3 Summary

A set of experimental flights were made to evaluate system performance. By analyzing the log files created by the drone on each flight made, it was possible to verify that the settings used are proper. The drone moves to the desired geographic coordinates, and the entire flight plan path is executed by the drone. It visited the desired waypoints and collected all data stored in the probes.

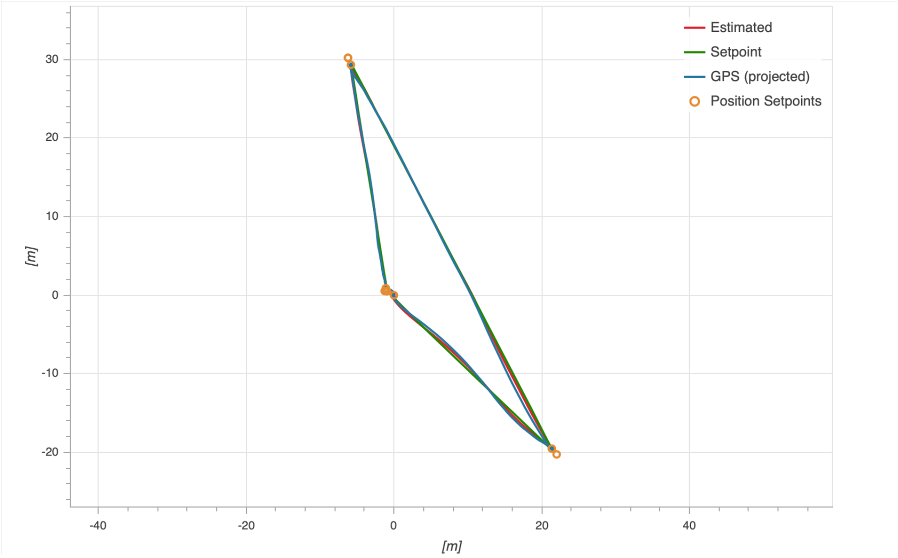


Figure 5.3: Estimated position of the drone

Examining the transmission time of data between Bluetooth devices, it turns out that the number of probes visited by the drone has a high impact on the duration of the mission and may limit its execution. This limiting factor of the system has not been resolved due to lack of time and was considered as future work.

Chapter 6

Conclusion and future work

6.1 Conclusion

The main goal of the project was to create a system that performs opportunistic data collection using an autonomous drone. Since agricultural data may be spread across fields, the drone autonomously navigates to them and collects the desired data. This system has been tested and designed for the use case of agriculture, but can be adapted for data collection in any outdoor environment.

This project offers some abstraction allowing the user to utilize the system in an easy and intuitive way, presenting a consistent behavior regardless of its user. Therefore, the user does not need to have a meticulous knowledge of the drone's operation to obtain satisfying results.

Several test flights were made to verify the system performance, and it was possible to conclude that the drone had good behavior when navigating to the desired points as well as collecting the stored data in the probes.

These test flights consist of drones navigation to two waypoints where probes were located. The drone navigates to these locations and collects the data stored in the probes. After visiting all probes, the drone returned to the original position, thus ending the mission.

In conclusion, the development of the system has revealed that it satisfies most of the proposed goals. Future additions may be done to turn the system more robust, efficient and complete. Nevertheless, despite the limitations, the system has an acceptable confidence level.

6.2 Future Work

Although the system accomplished most of the purposes, it is beneficial to improve some aspects and add new tools in the future. These tools would allow the system to become more efficient and more autonomous, not requiring significant user intervention.

- **More efficient Bluetooth transmission**

As mentioned, data transmission between Bluetooth devices takes up a lot of the drone's flight time. Sending all data at once or sending one by one without waiting for confirmation would be a way to make transfer faster and more efficient.

- **Data transfer optimization**

Creation of a tool to be used when the drone is in the probe's location. It changes the drone's position according to the signal strength of the probe. This way, the drone could approach the probe for more efficient and faster transmission.

- **Flight Plan optimization**

Development of a tool to automatically create a flight plan taking into account the location of the probes. The user, through an application on a smartphone, could indicate the location of the added probe. After adding all the probes, and using weather information, the tool creates a flight plan taking into account the direction and intensity of the wind, planning a more favourable route with fewer energy costs. This tool would also ensure that the drone has enough battery to return to its initial location.

- **Image capture**

Add an image capture module to the drone, making it possible to map the field and record the plants' evolution. With just one flight, the drone could collect the data from the probes and also collect images of the field for later analysis.

- **Survey flight**

Creation of a survey flight, where the drone travels the entire field, making a pattern flight to find all probes present in the field and record their locations. The locations would be used to create a flight plan. Applying this tool, the user does not need to inform the system of the location of the probes, making it more autonomous.

Appendix A

Plots from log file

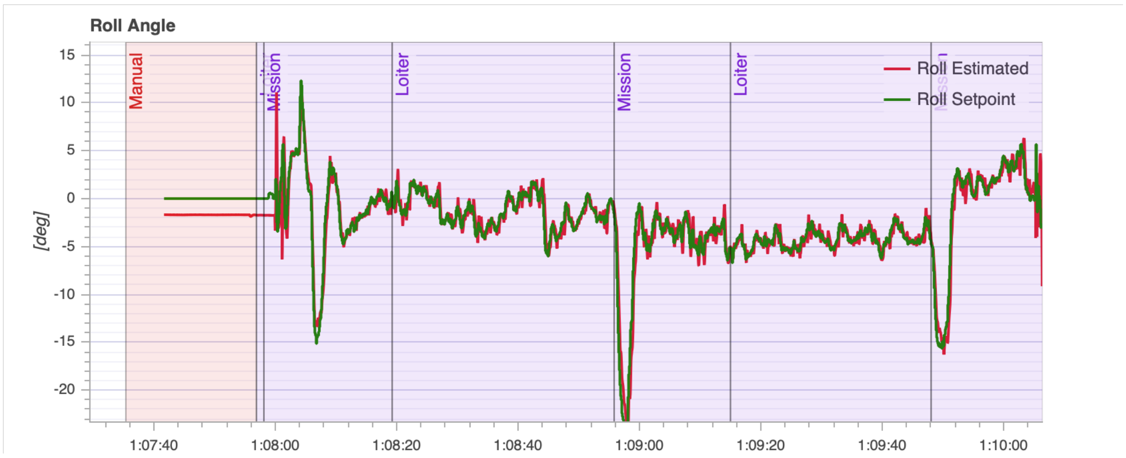


Figure A.1: Roll Angle Plot

Figure A.2: Pitch Angle Plot

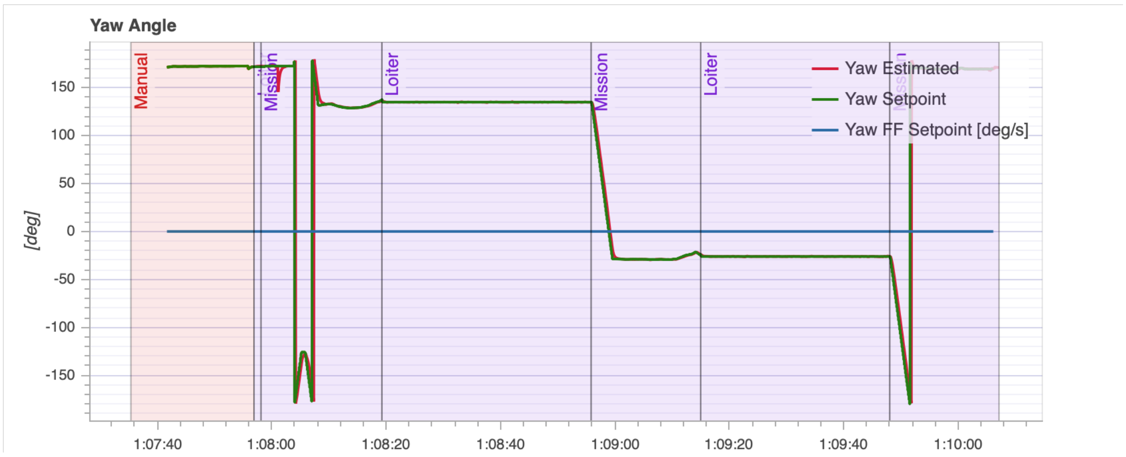


Figure A.3: Yaw Angle Plot

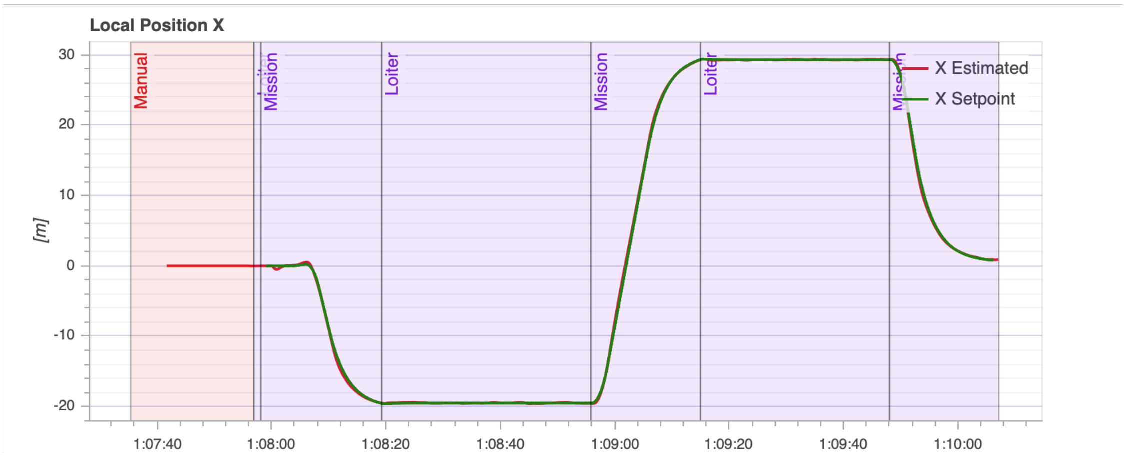


Figure A.4: Local Position X

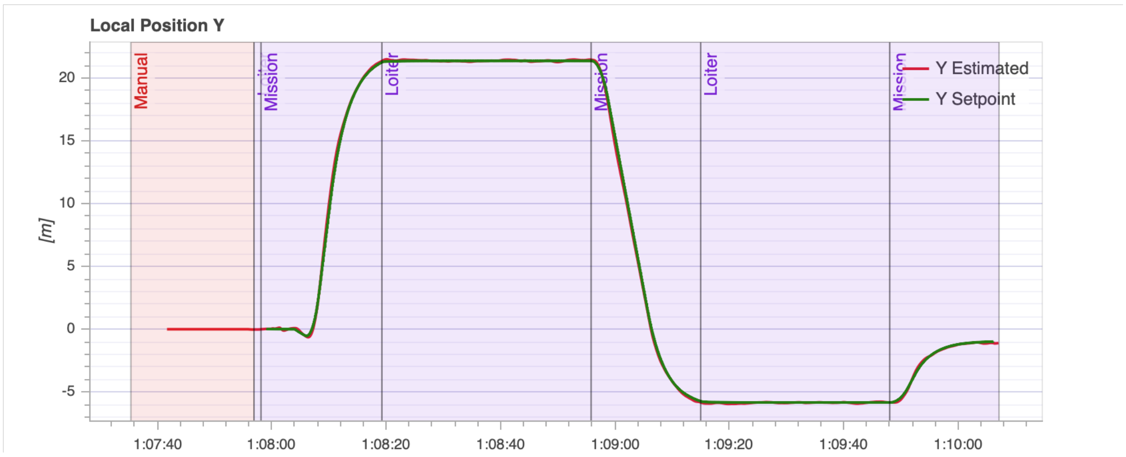


Figure A.5: Local Position Y

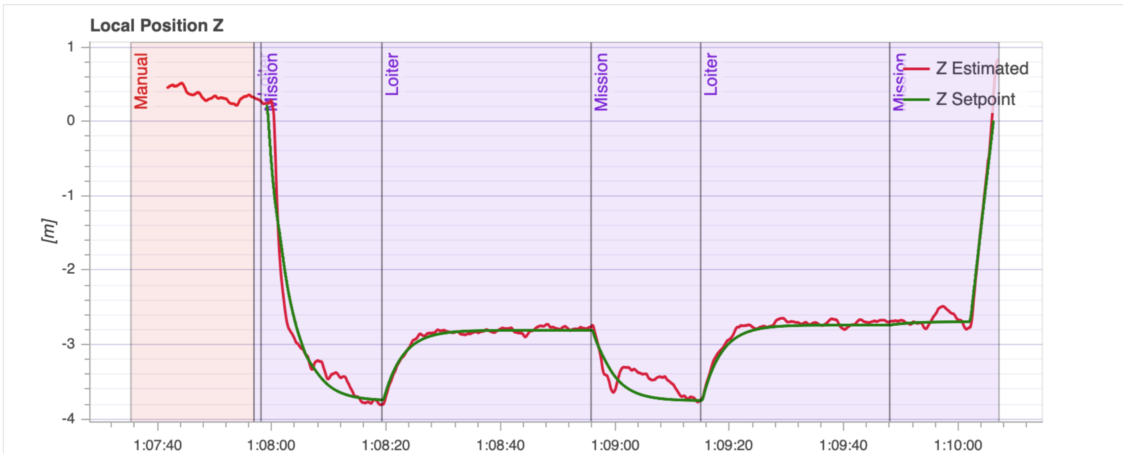


Figure A.6: Local Position Z

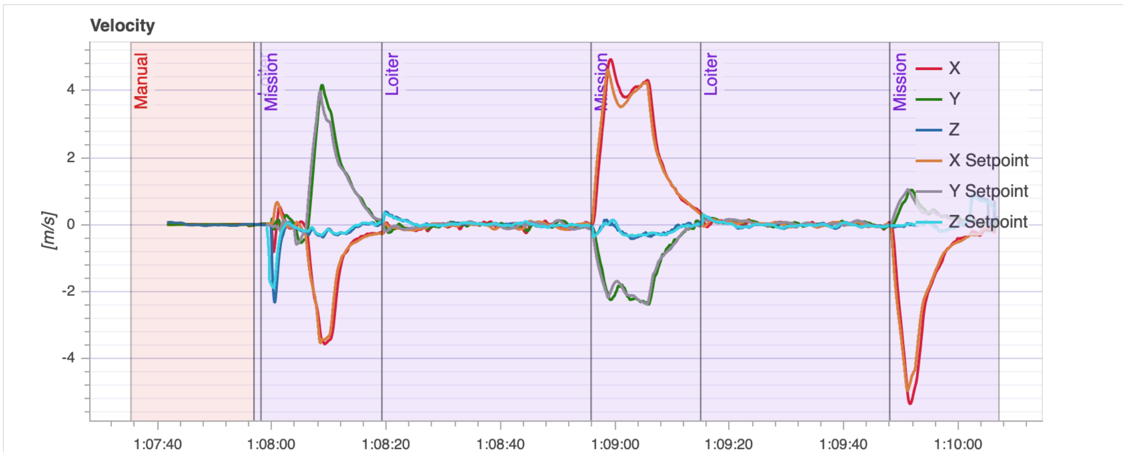


Figure A.7: Velocity

References

[1] A. Nawrat, K. Jędrasiak, K. Daniec, and R. Koteras, “Inertial Navigation Systems and Its Practical Applications,” *New Approach of Indoor and Outdoor Localization Systems*, Oct. 2012. [Online]. Available: https://www.intechopen.com/books/new-approach-of-indoor-and-outdoor-localization-systems/inertial-navigation-systems-and-its-practical-applications

[2] “Airframes Reference · PX4 User Guide.” [Online]. Available: https://docs.px4.io/master/en/airframes/airframe_reference.html

[3] “Aircraft Yaw Pitch Roll - The Best and Latest Aircraft 2018.” [Online]. Available: http://krotoku.com/aircraft-yaw-pitch-roll/

[4] “PX4 Architectural Overview · PX4 Developer Guide.” [Online]. Available: https://dev.px4.io/master/en/concept/architecture.html

[5] “ESP32-DevKitC Development Boards - Espressif Systems | Mouser Portugal.” [Online]. Available: https://pt.mouser.com/new/espressif/espressif-esp32-devkitc-boards/

[6] M. Kazin, “Drone operators for commercial use to quadruple by 2022: FAA,” Mar. 2018. [Online]. Available: https://www.foxbusiness.com/features/drone-operators-for-commercial-use-to-quadruple-by-2022-faa

[7] “Popular Uses for Drones.” [Online]. Available: https://www.dummies.com/consumer-electronics/drones/popular-uses-for-drones/

[8] B. Custers, Ed., *The Future of Drone Use: Opportunities and Threats from Ethical and Legal Perspectives*, ser. Information Technology and Law Series. T.M.C. Asser Press, 2016. [Online]. Available: https://www.springer.com/gp/book/9789462651319

[9] N. J. Stehr, “Drones: The Newest Technology for Precision Agriculture,” *Natural Sciences Education*, vol. 44, no. 1, pp. 89–91, Dec. 2015. [Online]. Available: https://dl.sciencesocieties.org/publications/nse/abstracts/44/1/89

[10] “FAO - News Article: 2050: A third more mouths to feed.” [Online]. Available: http://www.fao.org/news/story/en/item/35571/icode/

[11] S. Heble, A. Kumar, K. V. V. D. Prasad, S. Samirana, P. Rajalakshmi, and U. B. Desai, “A low power IoT network for smart agriculture,” in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, Feb. 2018, pp. 609–614.

[12] S. Jaiganesh, K. Gunaseelan, and V. Ellappan, “IOT agriculture to improve food and farming technology,” in *2017 Conference on Emerging Devices and Smart Systems (ICEDSS)*, Mar. 2017, pp. 260–266.

[13] P. Singh, P. Pawar, P. Takwale, and D. Karia, “Agriculture monitoring system AMS,” in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Aug. 2017, pp. 742–745.

[14] Fan TongKe, “Smart Agriculture Based on Cloud Computing and IOT,” *Journal of Convergence Information Technology*, vol. 8, no. 2, pp. 210–216, Jan. 2013. [Online]. Available: http://www.aicit.org/jcit/global/paper_detail.html?jname=JCIT&q=2598

[15] G. A. Kumar, M. Ramya, and C. Ram Kumar, “Wedding of robots with agriculture,” in *2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT’12)*. Coimbatore: IEEE, Jul. 2012, pp. 1–4. [Online]. Available: http://ieeexplore.ieee.org/document/6396101/

[16] S. S. H. Hajjaj and K. S. M. Sahari, “Review of agriculture robotics: Practicality and feasibility,” in *2016 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, Dec. 2016, pp. 194–198.

[17] R. Eaton, J. Katupitiya, K. W. Siew, and B. Howarth, “Autonomous Farming: Modeling and Control of Agricultural Machinery in a Unified Framework,” in *2008 15th International Conference on Mechatronics and Machine Vision in Practice*, Dec. 2008, pp. 499–504.

[18] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*. Scituate, MA, USA: Bradford Company, 2004.

[19] G. Amer, S. M. M. Mudassir, and M. A. Malik, “Design and operation of Wi-Fi agribot integrated system,” *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, pp. 207–212, 2015.

[20] Q. Feng, X. Wang, G. Wang, and Z. Li, “Design and test of tomatoes harvesting robot,” in *2015 IEEE International Conference on Information and Automation*, Aug. 2015, pp. 949–952.

[21] F. A. A. Cheein and R. Carelli, “Agricultural Robotics: Unmanned Robotic Service Units in Agricultural Tasks,” *IEEE Industrial Electronics Magazine*, vol. 7, no. 3, pp. 48–58, Sep. 2013.

[22] A. Vasudevan, D. A. Kumar, and N. S. Bhuvaneswari, “Precision farming using unmanned aerial and ground vehicles,” in *2016 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR)*, Jul. 2016, pp. 146–150.

[23] N. S. Naik, V. V. Shete, and S. R. Danve, “Precision agriculture robot for seeding function,” in *2016 International Conference on Inventive Computation Technologies (ICICT)*, vol. 2, Aug. 2016, pp. 1–3.

[24] D. Yallappa, M. Veerangouda, D. Maski, V. Palled, and M. Bheemanna, “Development and evaluation of drone mounted sprayer for pesticide applications to crops,” in *2017 IEEE Global Humanitarian Technology Conference (GHTC)*, Oct. 2017, pp. 1–7.

[25] “Agriculture Drones (Precision Agriculture with NDVI) | QuestUAV.” [Online]. Available: https://www.questuav.com/industries/agriculture-drones/

[26] “Agras MG-1s - Agricultural Solutions - DJI.” [Online]. Available: https://www.dji.com/pt/mg-1s

[27] “What Is A Fixed Wing Drone? | Advantages And Uses Of Fixed Wing Drones,” Sep. 2018. [Online]. Available: https://www.uavsystemsinternational.com/what-is-a-fixed-wing-drone-advantages-and-uses-of-fixed-wing-drones/

[28] “ADVANTAGES OF FIXED WING UAV THAN ROTARY WING | Clever Drone Maps.” [Online]. Available: https://www.cleverdronemaps.com/advantages-of-fixed-wing-uav-than-rotary-wing/?lang=en

[29] K. Agrawal and P. Shrivastav, “Multi-rotors : A Revolution In Unmanned Aerial Vehicle,” 2015.

[30] DroneDeploy, “Choosing the Right Mapping Drone for Your Business Part I: Multi-Rotor vs. Fixed Wing Aircraft,” Jun. 2017. [Online]. Available: https://blog.dronedeploy.com/choosing-the-right-mapping-drone-for-your-business-part-i-multi-rotor-vs-fixed-wing-aircraft-6ec2d02eff4

[31] 3DR, “3dr Pixhawk.” [Online]. Available: https://3dr.com/support/articles/3dr_pixhawk/

[32] ArduPilot, “Archived:APM 2.5 and 2.6 Overview — Copter documentation.” [Online]. Available: http://ardupilot.org/copter/docs/common-apm25-and-26-overview.html

[33] “CopterControl / CC3d / Atom Hardware Setup — LibrePilot/OpenPilot Wiki 0.1.4 documentation.” [Online]. Available: https://opwiki.readthedocs.io/en/latest/user_manual/cc3d/cc3d.html#getting-to-know-your-board

[34] “Intel® Aero Ready to Fly Drone.” [Online]. Available: https://www.intel.com/content/www/us/en/products/drones/aero-ready-to-fly.html

[35] “Quadcopter AR Drone 2.0 Elite Edition | Parrot Official.” [Online]. Available: https://www.parrot.com/global/drones/parrot-ardrone-20-elite-edition

[36] “Parrot AR.Drone 2.0 GPS Edition,” Nov. 2017. [Online]. Available: https://www.parrot.com/pt/en/drones/parrot-ardrone-20-gps-edition

[37] “DJI Mavic Pro & Mavic Pro Platinum – Every Creative Moment – DJI.” [Online]. Available: https://www.dji.com/pt/mavic

[38] L. Meier, “The History of Pixhawk.” [Online]. Available: https://auterion.com/the-history-of-pixhawk/

[39] ArduPilot, “Mission Planner Home — Mission Planner documentation.” [Online]. Available: http://ardupilot.org/planner/index.html#home

[40] QGroundControl, “Overview · QGroundControl User Guide.” [Online]. Available: https://docs.qgroundcontrol.com/en/

[41] X. Li, Y. Deng, and L. Ding, “Study on precision agriculture monitoring framework based on WSN,” in *Security and Identification 2008 2nd International Conference on Anti-counterfeiting*, Aug. 2008, pp. 182–185.

[42] K. A. Patil and N. R. Kale, “A model for smart agriculture using IoT,” in *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, Dec. 2016, pp. 543–545.

[43] “IoTiP.” [Online]. Available: https://www.aicos.fraunhofer.pt/en/our_work/projects/iotip.html

[44] A. Pereira and F. Nunes, “Physical Activity Intensity Monitoring of Hospital Workers using a Wearable Sensor,” Aug. 2018. [Online]. Available: https://eudl.eu/doi/10.4108/eai.20-4-2018.2276323

[45] Silva, Joana, Gomes, Diana, Gomes, Nunes, Nunes, Francisco, Moreira, Dinis, Alves, José, Pereira, Ana, and Sousa, Inês, “Position-independent Physical Activity Monitoring: Development and Comparison with Market Devices,” 2019.

[46] “nRF52832 - Bluetooth low energy System-on-Chip - nordicsemi.com.” [Online]. Available: https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832

[47] “Waspmote - Open Source Sensor Node for the Internet of Things | NB-IoT, Cat-M, ZigBee, Sigfox, LoRaWAN, 3g / 4g / GPRS | Libelium | Libelium.” [Online]. Available: http://www.libelium.com/products/waspmote/

[48] “Pycno Sensors.” [Online]. Available: https://pycno.co

[49] R. Tei, H. Yamazawa, and T. Shimizu, “BLE power consumption estimation and its applications to smart manufacturing,” in *2015 54th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, Jul. 2015, pp. 148–153.

[50] “ESP32 Overview | Espressif Systems.” [Online]. Available: https://www.espressif.com/en/products/hardware/esp32/overview

[51] “Overview · QGroundControl User Guide.” [Online]. Available: https://docs.qgroundcontrol.com/en/

[52] “Introduction · MAVLink Developer Guide.” [Online]. Available: https://mavlink.io/en/

[53] “jMAVSim Simulation · PX4 v1.9.0 Developer Guide.” [Online]. Available: https://dev.px4.io/v1.9.0/en/simulation/jmavsim.html

[54] “Flight Review.” [Online]. Available: https://logs.px4.io/