Modular ESC Motors Drone (III part)

Paolo Furia ID 239451
Dept. of Industrial Engineering
paolo.furia@studenti.unitn.it

***Abstract*—In recent years, the use of drones has increased in a wide range of applications. The need to quickly replace components, to quickly recharge batteries and to have variable geometry that adapts to all circumstances have become a major challenge in the field of drones. In this paper, work done by others was analysed, redone and improved upon in which an innovative method was presented to create a communication link between a Flight Controller (FC) and an Electronic Speed Controller (ESC) in a drone, using Ultra-Wide Band (UWB) technology to transmit control signals.**

In particular, the communication protocol of the receiver modules was improved, the receiver module was built and installed, and finally, simulative tests and practical flight tests were carried out, validating how this communication technology and mechanical design is a fundamental step forward in the development of efficient and reliable drones that have no geometric constraints

***Index Terms*—Variable drone geometry, modular ESC, UWB, quadrotor drone, SITL**

ACRONYMS

DC	Duty Cycle.
DRV	Driver Layer.
ESC	Electronic Speed Controller.
FC	Flight Controller.
FCC	Federal Communications Commission.
GPS	Global Positioning System.
HAL	Hardware Access Level.
IMU	Inertial Measurement Unit.
ISR	Interrupt Service Routine.
ITU-R	International Telecommunication Union Radiocommunication.
MCU	Microcontroller Unit.
MMA	Motor Mixing Algorithm.
PCB	Printed Circuit Board.
PDB	Power Distribution Board.
PID	Proportional–Integral–Derivative.
PPI	Programmable Peripheral Interface.
PWM	Pulse Width Modulation.
RF	Radio Frequency.
SITL	Software In the Loop.
SWD	Serial Wire Debug.
UAV	Unmanned Aerial Vehicles.
UWB	Ultra-Wide Band.
VTOL	Vertical Take-Off and Landing.
WPAN	Wireless Personal Area Network.

I. INTRODUCTION

DRONES, also known as Unmanned Aerial Vehicles (UAV), have become increasingly popular in recent years due to their versatility and wide range of applications such as inspection, agriculture, entertainment, package delivery,etc.. The development of microelectro-mechanical systems (MEMS), sensors, fabrication, navigation methods, remote control capabilities, and power storage systems has enabled the design and manufacture of UAVs in various sizes, configurations, and performances. Drones are usually controlled by a flight controller that manages the drone's flight by adjusting the speed of its motors. When referring to drones, it is common to refer to the category of multirotor Vertical Take-Off and Landing (VTOL) vehicles of small to medium size that are remotely controlled and have more than two rotors.

A UAV consists of the following components:

- **Airframe:** is the physical structure that provides the framework for attaching various components, determines the UAV's shape, size, weight, and aerodynamic properties.
- **Flight Controller (FC):** uses sensors such as accelerometers, gyroscopes, magnetometers, and Global Positioning System (GPS) to estimate the UAV's attitude and position to stabilize the drone, maintain altitude, and control flight maneuvers regulating the motor speed.
- **Power Source:** drones are powered by rechargeable batteries, usually lithium polymer (LiPo) or lithium-ion batteries. The power source supplies electricity to the propulsion system, onboard electronics, and other components.
- **Payload:** refers to any equipment or devices carried by the UAV, such as cameras, external sensors, delivery packages, or other specialized equipment depending on the specific application of the UAV.
- **Communication System:** drones often utilize Radio Frequency (RF) transmitters and receivers for communication with ground control stations or remote pilots. This enables real-time control, telemetry data transmission, and video streaming from onboard cameras.
- **Propulsion system:** provides the necessary thrust to enable the UAV to fly and perform its intended functions. It consists of the Electronic Speed Controller (ESC), motors, and propellers.

A. UWB Technology

Ultra-Wide Band (UWB) is a short-range wireless communication protocol that operates by sending radio waves of short duration pulses over a wide spectrum of frequencies ranging from 3.1 to 10.6 GHz leading to a 7.5GHz bandwidth. The Federal Communications Commission (FCC) and the International Telecommunication Union Radiocommunication

(ITU-R) has defined an UWB device as any device with a $-10dB$ fractional bandwidth, greater than 20% or occupying at least 500MHz of the spectrum [1]. Designed for Wireless Personal Area Network (WPAN) the main advantages of the UWB technology are: high data rate, reduced complexity, low power consumption, narrowband interference attenuation, as well as multiple transceiver architectures for different ranges. Thanks to the low power spectral density, UWB systems do not interfere with other radio frequency signals and have good penetration properties through different materials, improving the coverage of the UWB radios. Furthermore, high accuracy and good multipath performance are possible due to the short pulse duration that UWB systems have. UWB is used in a variety of applications including high-speed data transfer, indoor positioning, radar imaging, smart factory, vehicle tracking and wireless sensor networks. It is also commonly used in applications that require low power consumption and high levels of security. [2][3]

B. Previous work, goals and problems

In order to optimise and continue the previous project, it was necessary to go over and test the entire previous work again. The following is a detailed summary of the previous work, with the addition of some useful points for possible improvement and a better understanding of the project as a whole.

In previous work, an innovative method based on Ultra-Wide Band (UWB) technology was used to make the Flight Controller (FC) and Electronic Speed Controller (ESC) communicate. The development of the communication system was the main focus. An initial prototype of the drone was built, followed by initial flight tests and simulations that demonstrated the effectiveness of the method.

The multi-rotor drone consists of:

- **Master module** consisting of different components: Flight Controller (PixHawk4), a radio receiver, a battery, a power distribution board and a UWB transmitter module (DWM1001 module) as shown in Figure 1.
  The radio receiver receives the commands from the remote control and transfers them to the Flight Controller. The latter uses data from the Inertial Measurement Unit (IMU), which includes sensors such as accelerometers, gyroscopes, magnetometers, barometers and GPS, to perform various actions such as stabilization and navigation to a specific location adjusting the motor speeds through a PID control algorithm that sends PWM signals at a fixed frequency (400 Hz) to the ESCs. Those signals are not sent directly to the ESC, instead they are collected by the Microcontroller Unit (MCU) of the transmitter module and then transmitted via UWB to the receiver modules of the slaves' modules. The messages sent includes the address of each receiver, followed by the Duty Cycle (DC) that needs to be reproduced. The modules are synchronized so the rising edges of every channel occur at the same time. The falling edges are of course dependent on the duty cycle DC of the specific channel. The minimum DC that correspond to 0 rpm is 40% (i.e. minimum duration of 1[ms]) and a maximum of 80% (minimum duration of 2[ms]) which corresponds to full throttle as shown in Figure 2.

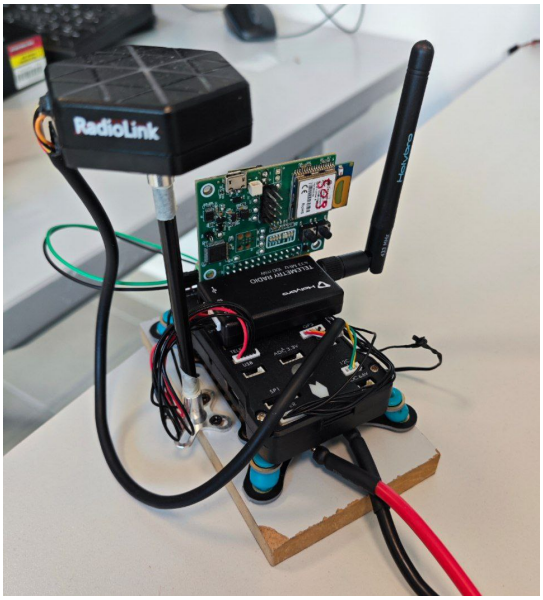


Fig. 1: Master Module

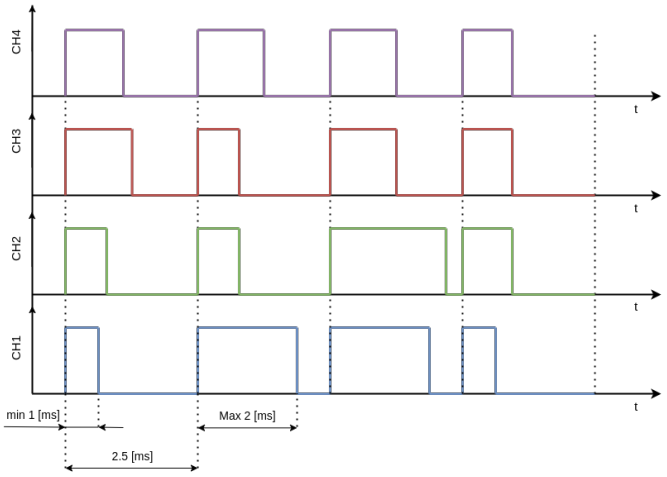


Fig. 2: Quadcopter FC PWMs example

The architecture proposed of this innovative method based on UWB is depicted in Figure 3.

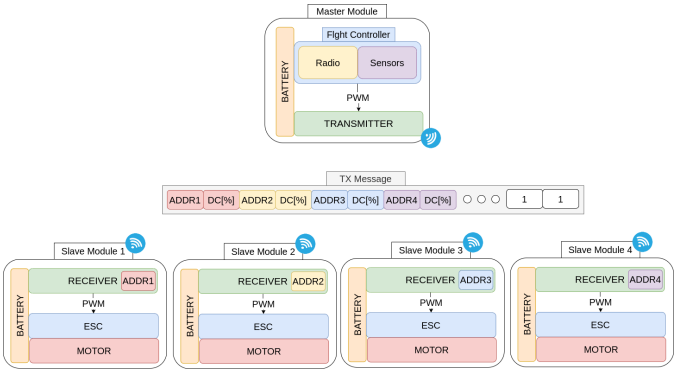


Fig. 3: UWB wireless multirotor motor drone architecture proposed

The PWM signal generated by the Flight Controller is received by the MCU of the transmitter module through the implementation of a Programmable Peripheral Interface (PPI) that operates at a frequency of 16[MHz]. PPI allows to trigger a *task* (e.g. independent sections of code that perform specific functions or operations) in one peripheral as result of an *event* (e.g. signals or notifications that occur within the system, triggering specific actions or behaviours that can be generated by various sources, such as hardware peripherals, timers, interrupts, or other tasks) occurring in another peripheral while excluding the CPU from these operations. A PPI block diagram is shown in Figure 4.

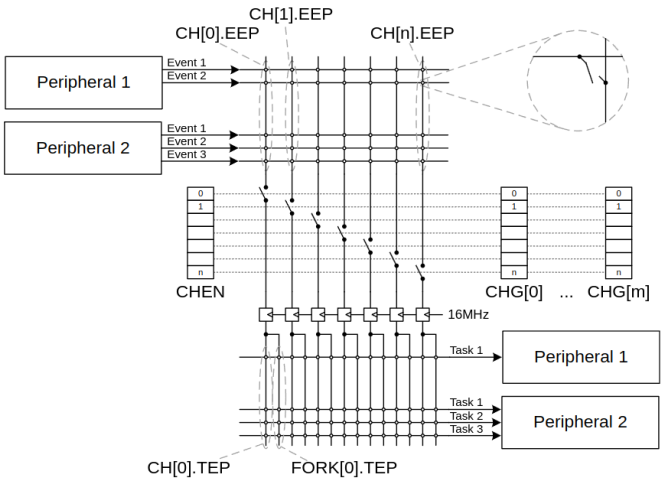


Fig. 4: PPI block diagram [4]

In our work each PWM channel is assigned to a dedicated timer which works in *input capture* mode. Interconnections are established between an *event*, generated by the detection of an edge (rising or falling) of the input pin connected to the FC through PPI, and the timer *task* that immediately saves (without CPU intervention) the timer counter (`CNT`) to the Capture Compare register (`CC[0]`) of the associated timer. To retrieve the duty cycle information is therefore necessary to differentiate between the times of the rising and falling edges, for this purpose, when an event is triggered, the CPU enters a specific Interrupt Service Routine (ISR) designed to distinguish the edge and save the Capture Compare register `CC[0]` in memory. The duty cycle of the signals is then computed using a simple relation:

$$DC = \frac{T_{fall} - T_{rise}}{T} \tag{1}$$

where T_{fall} and T_{rise} are the times at which the falling or rising edge occur and T is the period of the signal.

- **Slave module** are the ones that can be added in arbitrary number to the airframe which consist of a UWB receiver module (DWM1001), a battery an ESC and a motor with its propeller. The receiver reads the message sent by the receiver, looks for its address and read the desired DC that has to be replicated.
  In order to achieve this, the MCU's output GPIO is connected to its `TIMER[0]`, which is configured to generate a square wave of fixed frequency. The timer is triggered by an initialization signal sent by the transmitter, which serves as a sync signal for all transmitters. To adjust the DC, the CPU modifies the value of the `CC[0]` register, which changes the duty cycle.

In the previous work, the delay introduced by the acquisition technique through PPI of the PWM signal was measured using the PicoScope 5444D MSO oscilloscope. In addition, the overall delay introduced by this system is characterised.

It was found that the time required for the MCU of the transmitter module to acquire PWM signal (e.g. from the moment an event occurs in the GPIO to the moment the timer counter `CNT` is saved in the `CC[0]` register) via PPI was 23[us] for the first channel and 15[us] for each additional channel. Consequently, the time required to save the PWM data of 4 modules was 68[us]. This time difference is caused by the fact that the PWM signals are sent synchronously by the FC and consequently the CPU of the transmitter receives several simultaneous interrupts. Since it can execute one Interrupt Service Routine (ISR) at a time, the requests accumulate and the CPU takes less time to serve the accumulated requests. Since the FC operates at a frequency of 400[Hz] with a period of 2.5[ms] and measurements had to be completed by reading each channel's `CC[0]` register within 20% of the period that corresponds to 500[us], the maximum number of modules was:

$$\text{max n. channels} = \frac{500[us] - 23[us]}{15[us/channel]} + 1 \approx 31 \tag{2}$$

Finally, the analysis focused on the delay introduced by the communication system, specifically the time elapsed between the generation of the PWM signal by the FC and its replication by the transmitter of the slave module.

Figure 5 shows one PWM channel of the FC with period *T-FC* and the replicated PWM on the receiver side with period *T-REC*, furthermore auxiliary signal (scaled down by a factor of ten) are used for debugging purposes.

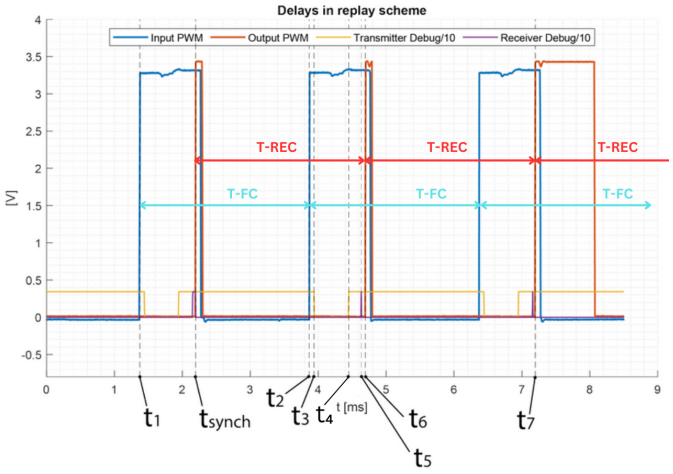


Fig. 5: Delay in the reply scheme

As can be seen, the PWM of the receiver module start when it receives the synchronization signal t_{synch}, which happens before the falling edge of the input PWM can be detected. Thus, the initial phase of the generated PWM (e.g. the period T-REC) commences from a random value distinct from the one generated by the Flight Controller.

Meanwhile, in the first T-FC period, the MCU of the transmitter module acquires the first PWM signal.

Once the T-FC period is over, the transmission of the message by the transmitting module begins with a delay of $\Delta t = t_3 - t_2 = 72[us]$, this delay is caused by the reading of the `CC[0]` registers of the timers and their storage in the memory.

The time delay $\Delta t = t_4 - t_3 = 509[us]$ was captured by lowering the debug pin of the transmitting module when message preparation begins and raising it once the message has been sent.

The message arrives at the receiving module with a delay that depends partly on the distance from the transmitter and partly on the analysis of the message itself. This delay $\Delta t = t_5 - t_4 = 189[us]$ was measured by raising the pin of the receiving module.

Finally, there is a delay $\Delta t = t_6 - t_5 = 59[us]$ from the beginning of the next period.

As can be seen from the Figure 5, even if the message is received and analysed before the receiver starts the second T-REC period, the duty cycle is not updated. In order to see the duty cycle initially collected (in the first T-FC period) from the receiver side, it is necessary to wait for the third period of the T-REC receiver module. This means that the duty cycle is replicated with more than two periods of delay, precisely $\Delta t = t_7 - t_1 = 5831[us]$

A flight test using the same airframe geometry was conducted, during which the hover manoeuvre was executed. It was found that the drone vibrates significantly more with the wireless ESC connection than with the wired configuration throughout the entire flight period. Excessive vibration can cause various issues, such as reduced flight efficiency and duration, position estimation failures leading to in-flight escapes, or inaccurate vehicle adjustments resulting in degraded flight performance.

Possible causes of this instability include delayed communication between the flight controller and the ESC module, or asynchronization of the ESC modules. These factors can introduce instabilities at certain levels.

C. Custom boards

While a commercial PCB (DWM1001-dev) had previously been used, in order to minimise space and make more GPIO ports available so that they were easily accessible and could be connected to the FC, it was decided to make a custom Printed Circuit Board (PCB) that mounted the UWB module (DWM1001c). The latter is powered by a 4s battery (14.8V) and was equipped with a voltage regulator with a conditioning circuit so that it could power the ESCs (in the case of the slave modules) and the flight controller (in the case of the master module). Two XT60 connectors (one male and one female) were therefore installed to facilitate the connection and eventual replacement of components. For this purpose, it was decided to use a switching regulator (LM53603) capable of achieving an output voltage of 3V3 by choosing a suitable feedback resistor. The dimensions of the board are 75×38.7×1.6 mm (L×W×H). A diagram of the board is shown in Figure 6.

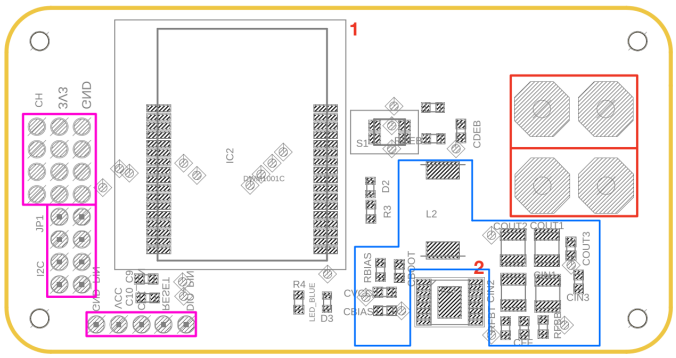


Fig. 6: Custom PCB. Where: (1)=UWB module, (2)=Voltage converter and conditioning circuit, (red square)=Connector for the battery and for the ESC, (pink squares)=GPIO Pins, (S1)= Reset button

As can be seen from the above figure, 12 pins were used in the top left-hand corner so that it could be connected to the FC. Each line represents a channel, which is connected to a specific GPIO of the MCU (`SPIS_MISO`, `SPIS_MOSI`, `GPIO_8`, `GPIO_15`), which correspond to ports `P0.07`, `P0.06`, `P0.08`, `P0.15` (later used in the code). Finally, five lines (bottom of the figure) were dedicated so that the appropriate firmware could be loaded using a J-Link programmer via Serial Wire Debug (SWD). The pins of the PCB, from left to right in the figure are `GND`, `VCC`, `CLK`, `RESET`, `DIO` which must be connected to the J-Link debugger pins `GND`, `VTref`, `SWCLK`, `RESET`, `SWDIO` respectively.

II. Description of the Project

The objective of this project is to continue the study of a modular ESC motor drone. In particular, the following issues were addressed:

- In Section III a description of the components used is made. Finally, the design and the installation of the slave modules so that they can be effectively used on any object is addressed.
- In Section IV a Software In the Loop (SITL) simulation is conducted to analyse the impact of motors positioning and motor delay on flight stability.
- In Section V the receiver delay is reduced improving the communication code performances.
- In Section VI a flight test was conducted by mounting the modules on a empty cardboard box, the results were then analysed.

III. Slave module

A. Hardware components used

The hardware components used with their main parameters are listed below.

- **Battery** The battery chosen is a Lithium polymer (LiPo) battery which is commonly used in drones due to their flexibility and lightness compared to lithium-ion (Li-ion) batteries. They offer greater energy storage and a longer cycle life. Each cell in a LiPo battery has a nominal voltage of 3.7V, with a safe operating range between 3.0V and 4.2V. Exceeding 4.2V can cause fire risks, while dropping below 3.0V can lead to irreversible damage.
  LiPo batteries can have multiple cells, denoted by the 'S' rating, indicating the number of cells. For instance, a 4S battery has four cells with a total nominal voltage of 14.8V. It's important to ensure all cells are balanced to avoid damage or reduced performance.
  Higher cell count batteries can offer more power but increase weight and cost. Capacity, measured in milliampere-hours (mAh), determines the current draw before depletion. Larger capacity can extend flight time but increases weight and size, impacting efficiency and overall performance. The C rating indicates the maximum current a LiPo battery can deliver without damage, calculated as the product of capacity and C rating. Proper balance in capacity, weight, and power is essential for optimal drone performance.

TABLE I: Battery specs

Turnigy Nano-Tech Plus 1400mAh 4S 70C Lipo Pack	
Capacity	1400mAh
Voltage	4S1P / 4 Cell / 14.8V
Discharge	70C Constant / 140C Burst
Weight	183g
Dimensions	74x35x39.3mm
Balance Plug	JST-XHR
Discharge Plug	XT60

Fig. 7: Battery used: Turnigy Nano-Tech Plus 1400mAh 4S 70C Lipo Pack

- **Motor** The most crucial factors to be considered when selecting the optimal motor are its weight, power, thrust, efficiency, torque, and response (RPM changes). A general guideline is that the maximum thrust produced by all motors should be at least double the total weight of the quadcopter. Therefore, it is of paramount importance to accurately estimate the total weight of the drone. Consequently, the thrust-to-weight ratio (or power-to-weight ratio) is a pivotal parameter. A higher thrust-to-weight ratio confers greater agility and acceleration upon the quadcopter, yet it can also render it more challenging to control. High torque motors facilitate rapid RPM changes and faster response times, which in turn result in less propeller wash oscillation and more responsive actions. The motor torque is dependent on a number of factors, including the stator size (volume), the materials of the magnets and the quality of the copper windings, as well as the motor construction (e.g. air gap, number of poles). The KV value indicates the number of revolutions per minute (rpm) a motor can achieve when 1V (one volt) is applied without any load (e.g. a propeller) attached to the motor. Higher KV motors attempt to rotate the propeller at a faster rate, resulting in increased thrust and power consumption (more current is drawn). Higher KV motors have shorter windings and lower resistance, which reduces the maximum voltage rating and increases the current draw for the motor and propeller combination. Typically, larger propellers are paired with low KV motors, while smaller, lighter propellers work better with high KV motors. The torque of a motor is not directly affected by its KV rating. However, the torque constant is influenced by the KV rating, which determines the amount of current required to generate a specific torque. Higher KV motors exhibit a higher torque constant, indicating that they require more current to produce the same torque as a lower KV motor. The number of poles has a direct impact on motor performance.
  In a nutshell more poles lead to a smoother performance while fewer poles lead to an increased power. Since drone motors are typically 3-phase, the pole configuration must be a multiple of 3.
  Two different motors were used to carry out the project. The respective parameters are shown in the following tables.

TABLE II: Motor specs

AX-2810Q-750KV	
Kv	750 rpm / V
Nominal power	444W @ 18.5V, 333W @ 11.1V
Weight	70g

TABLE III: Motor specs

Turnigy Multistar 4822-690Kv	
Kv	690 rpm / V
Max Surge Watt	320W@14.8V
Operating Current	<16A
Max operating current	22A
Weight	95g

(a) AX-2810Q-750KV

(b) Turnigy Multistar 4822-690Kv

- **ESC** The Electronic Speed Controller (ESC) is a crucial component of drone performance, as it is responsible for controlling the variable speed of motors. It is powered by direct current (DC) and takes motor signals from the Flight Controller (FC), providing three-phase alternating current to power the motor. In order to select the appropriate ESC, it is necessary to ensure that it is compatible with the battery's voltage and can handle the current draw of the chosen motor and propeller. The ESC current rating indicates the maximum current it can handle without damage. It is important to note that this is not the current pushed to the motors. Most ESCs have an amp rating that is more than sufficient. There are two current ratings for an ESC: continuous and burst. The continuous current rating signifies the constant current the ESC can safely manage, while the burst current rating represents the maximum current the ESC can handle for short periods, typically less than 10 seconds. The defining characteristic of an ESC is its firmware. The oldest and most renowned open-source ESC firmware is SimonK, followed by BLHeli. Another crucial aspect is the protocol employed by ESCs that determines the speed of the motor signal between the FC and the ESC. The protocols commonly used, from the most archaic to the most recent, are Standard PWM, OneShot, Multishot, Dshot, ProShot.

TABLE IV: ESC specs

Turnigy MultiStar BLheli_32 ARM 51A	
Constant Current	51A
N. of cells	2-6S Lipoly
Voltage	8.4∼25.2V
BEC	Opto
Timing	Auto
Frequency	48MHz
MCU	Arm Cortex-M0
Dimensions	36x21x5mm
Weight	17.2g

Fig. 9: ESC used: Turnigy MultiStar BLheli32 ARM 51A

- **Propellers** Without going into the details of aerodynamics, what most characterises propellers are the propeller length, propeller pitch and number of blades. The length of a propeller is determined by the diameter of the disc it creates when spinning, or the distance between the tips of a two-blade propeller. Propeller pitch refers to the distance a propeller travels per revolution and is measured in inches. A higher pitch propeller moves more air per revolution, which can generate more thrust when the aircraft is travelling at high speeds. Higher pitch can also create turbulence and propeller wash, which can affect the performance of the aircraft. It also rotates more slowly, which can make the aircraft less responsive. On the other hand, a lower-pitch propeller is more responsive and can spin up and down more quickly, improving manoeuvrability. Adding blades increases the surface area and therefore produces more thrust. This is similar to making the propeller longer, except you can fit it into a smaller disc area. Increasing the number of blades improves the grip in the air, but also makes it less efficient and puts more stress on the engine. Two blades are more efficient as they create less drag and draw less current, making them ideal for long range flying. As said, propeller size is given in imperial inches (1"=2.54cm) and are described with the following format: $L \times P \times B$ where L is the length, P is the pitch and B is the number of blades.
To estimate the maximum load a single slave module could lift a single slave module was taken into account, while on a mathematical level the following formula to first calculate the static thrust (T) has been employed [5] [6].

$$T = A \cdot \text{RPM} \cdot \frac{d^{3.5}}{\sqrt{\text{pitch}}} \cdot (B \cdot \text{RPM} \cdot \text{pitch} - V_0) \quad [N] \quad (3)$$

where $A = 4.392399 \times 10^{-8}$ and $B = 4.233333 \times 10^{-4}$ are two constants; RPM is the propeller rotations/min;

pitch refers to the distance a propeller travels during one revolution, and it's measured in inches; d is the propeller length; V_0 is the propeller forward airspeed measured in m/s that for our case is equal to zero because we are in the hovering case.

Two different types of propellers were used:

1) A 10×4.5×2 (length×pitch×#blades) propeller, that is paired with the motor *AX-2810Q* that has $KV = 750[rpm/V]$. Therefore, the max RPM is $750 \cdot 14.8 = 11100[rpm]$. It should be noted that when a propeller is mounted on the motor, the RPM decreases significantly due to air resistance. Hence, what we obtain applying Equation 3 is a value of $T = 15.368$[N], which correspond to a safety theoretical load of $1.567/2 \approx 0.7833$ [Kg] per slave module.
2) A 13×5×2 (length×pitch×#blades) propeller, that is paired with the motor *Turnigy Multistar 4822* that has $KV = 690[rpm/V]$. Therefore, the max RPM is $690 \cdot 14.8 = 10212[rpm]$. What we obtain applying Equation 3 is a value of $T = 34.35$[N], which correspond to a safety theoretical load of $3.5/2 \approx 1.75$ [Kg] per slave module.

B. Slave module design and installation

In the previous work, the drone was flown with a fixed geometry, which could not be modified as shown in Figure 10.

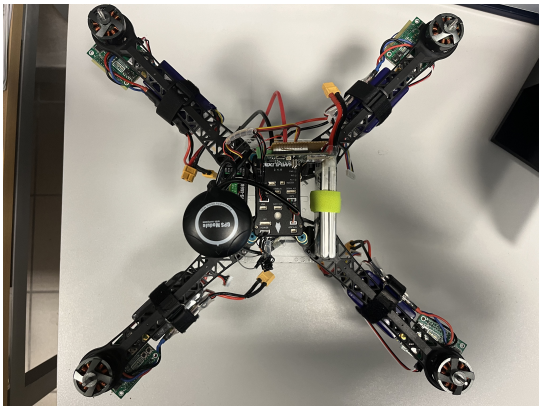


Fig. 10: Fixed Frame wireless motor drone

We then created a module docking system so that the drone could be docked to any object and thus have a modifiable geometry.

The method designed for attaching the slave module to any object is that of a c-clamp, which is a simple but very effective tool used for holding or fixing objects together. The clamping is done by means of a threaded screw and a knob (or butterfly nut), allowing the clamping pressure to be precisely controlled and adapted to different workpiece thicknesses. The main body and underlying plate are connected via two M8 rails and an M6 threaded rod. High-grip swivel pads can be used between the two plates to avoid damaging the object and preventing it from slipping. The prototype was printed using ABS, which is optimal for functional prototypes as it is a tough and durable material, resistant to heat and impact [7]. A honeycomb grill design was decided to be used to avoid excessive overheating. As shown in Figure 11 it consists of a compartment for inserting the necessary electronics such as the battery, ESC and switch, the latter being closed by means of a cap on which the receiver module is placed. The first model prototype had the following dimensions $100 \times 61.2 \times 51.5$ (L×W×H)[mm]. The total wight of the module, including the steel bars and the electronics components, is approximately 600[g]. The first slave geometry is shown in Figure 11

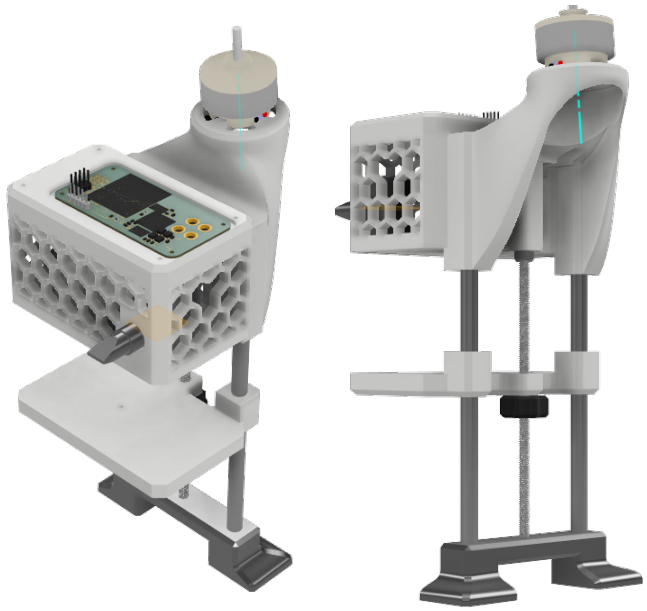


Fig. 11: Slave module geometry CAD model

However, the first prototype created has certain advantages and disadvantages. Among the advantages, we have the low weight and the small amount of space used. While the main disadvantage was found to be the difficulty of inserting the components inside the box with consequent strain on the cables and damage to the components during removal and insertion, as Figure 12 shows. It was therefore decided to create a second, more spacious prototype with dimensions 110×68×59[mm] and weight 645 [gr], furthermore a partition was also added to separate the battery from the ESC, thereby optimising heat dissipation and component layout. The new receiver module is shown in Figure 12. This proved to be an excellent compromise between ease of component insertion (and extraction) and space occupied.

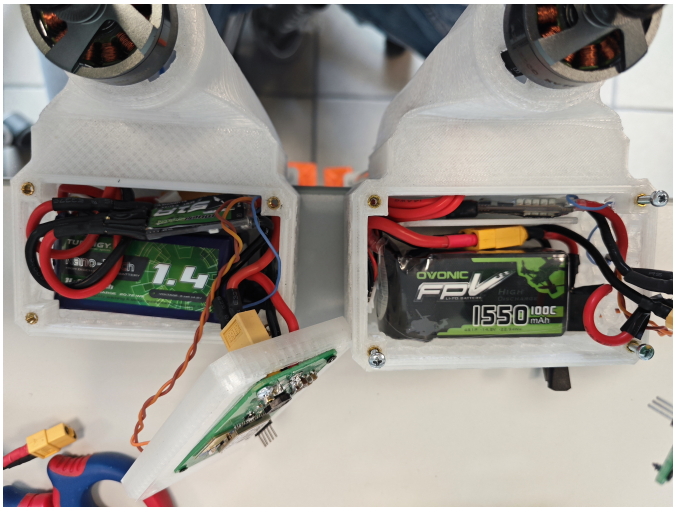


Fig. 12: (left) $100 \times 61.2 \times 51.5$[mm] old slave module, (right) $110 \times 68 \times 59$[mm] new slave module

IV. The Impact of Slave Module Positioning on Flight - SITL simulation

In order to understand how the geometry of the drone affects real flight testing, a Software In the Loop (SITL) simulation was performed. The PX4 SITL simulation works by running the PX4 flight stack on a computer, which communicates with a simulated vehicle in a virtual environment (Gazebo). The simulator provides synthetic sensor data to PX4, mimicking the inputs that would come from real sensors on an actual vehicle, PX4 processes this data and sends control commands back to the simulator, as if it were controlling a real vehicle, the simulated vehicle responds to these commands within the virtual environment, and the cycle continues, allowing for real-time interaction and testing.

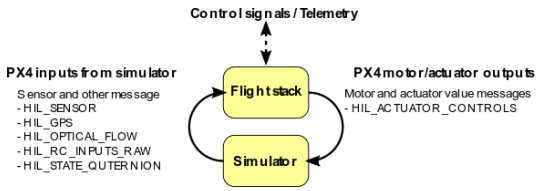


Fig. 13: SITL Simulation scheme

Four simulations were performed:

1) Drone where the motors are positioned so as to create a rectangular geometry $[(x_i, y_i)] = [(0.275, 0.1975), (-0.275, -0.1975), (0.275, -0.1975), (-0.275, 0.1975)]$ [m] as shown in Figure 14. Furthermore, the motor latency is null.

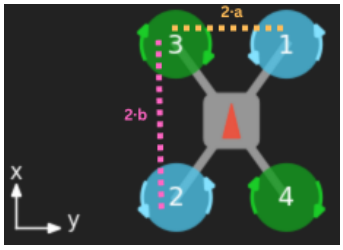


Fig. 14: Motor SITL position

2) Drone where the motors are positioned so as to create a rectangular geometry $[(x_i, y_i)] = [(0.275, 0.1975), (-0.275, -0.1975), (0.275, -0.1975), (-0.275, 0.1975)]$ [m] and the motor latency is considered equal to $6ms$.
3) Drone where the motors are positioned so as to create a rectangular geometry that is twice the first one $[(x_i, y_i)] = [(0.55, 0.395), (-0.55, -0.395), (0.55, -0.395), (-0.55, 0.395)]$ [m] and the motor latency is null.
4) Drone where the motors are positioned so as to create a rectangular geometry that is twice $[(x_i, y_i)] = [(0.55, 0.395), (-0.55, -0.395), (0.55, -0.395), (-0.55, 0.395)]$ [m] and the motors have a latency equal to 6ms.

The motors delay was simulated through the use of the previously developed FIFO buffer. It is important to emphasise that the inertia of the drone was not changed during the execution of the tests. In fact, this test is intended to emphasise how the positioning of the motors, which are assumed not to affect the total inertia of the drone, changes the flight stability of the drone. All the simulation performs the same planned mission that consists on:

1) Take off to an altitude of 10 meters
2) Remain at that height for 10 seconds
3) Land

Not taking into account the initial part where the drone takes off and the final part where the drone lands, in general what emerges from the simulations performed is that as the geometry of the prototype increases, i.e. as the distance of the motors from the center of the drone increases, the delay introduced by the implemented communication impacts more on the stability of the drone. Keeping a rectangular configuration of 0.275×0.1975[m] the introduction of a delay of 6 ms does not cause big differences. By introducing instead the same delay on a rectangular configuration twice as long as the previous one the latter has a greater effect, the drone in fact vibrates noticeably as can be seen from Figure 19d. One possible interpretation of this effect is that the force applied by the motor causes a greater momentum with respect to the center of mass of the drone being at a greater distance than in the previous configuration, the FC then, because of the delay, fails to correct in time the control previously performed. This may be due to how the PID controller weights are set affecting the performance of the Motor Mixing Algorithm (MMA). In fact, a problem that commonly occurs with a PID control when controller saturation limits are reached is known as

integral windup which can cause lengthy oscillations instead of settling, making the system unstable.
Therefore, it is important to emphasise that as the size of the drone increases, a delay in motor implementation has a greater impact. It is therefore crucial to reduce the delay introduced by the new communication system as much as possible.

From a mathematical point of view, without going into the details of the drone's dynamics derivation, practically we are changing the variables in red of the following non-linear dynamics of a generic x-quad rotor drone that include gyroscopic moments and additional inertial contributions. The equation of motion is derived following the nomenclature depicted in Figure 15.

$$
\begin{aligned}
\dot{u} &= \omega_z v - \omega_y w - \frac{K_{Dx}}{m}\left|u - v^b_{w,x}\right|\left(u - v^b_{w,x}\right) + g\sin\theta \\
\dot{v} &= \omega_x w - \omega_z u - \frac{K_{Dy}}{m}\left|v - v^b_{w,y}\right|\left(v - v^b_{w,y}\right) + \\
&\quad - g\cos\theta\sin\phi \\
\dot{w} &= \omega_y u - \omega_x v - \frac{K_{Dz}}{m}\left|w - v^b_{w,z}\right|\left(w - v^b_{w,z}\right) + \\
&\quad - g\cos\phi\cos\theta + \frac{\sum_i F_{T,i}}{m} \\
\dot{\omega}_x &= \frac{I_{yy} - I_{zz}}{I_{xx}}\omega_y\omega_z + \frac{a}{I_{xx}}\left(F_{T,1} + F_{T,4} - F_{T,2} - F_{T,3}\right) + \\
&\quad - \frac{I_p\left(4\omega_z + \sum_i \omega_{p,i}\right)\omega_y}{I_{xx}} \\
\dot{\omega}_y &= \frac{I_{zz} - I_{xx}}{I_{yy}}\omega_x\omega_z + \frac{b}{I_{yy}}\left(F_{T,3} + F_{T,4} - F_{T,1} - F_{T,2}\right) + \\
&\quad + \frac{I_p\left(4\omega_z + \sum_i \omega_{p,i}\right)\omega_x}{I_{yy}} \\
\dot{\omega}_z &= \frac{I_{xx} - I_{yy}}{I_{zz} + 4I_p}\omega_x\omega_y + \frac{\sum_i \tau_{D,i}}{I_{zz} + 4I_p} - \frac{I_p\sum_i \dot{\omega}_{p,i}}{I_{zz} + 4I_p}
\end{aligned}
\tag{4}
$$

where u, v, w are the linear velocity components along the body-fixed x, y, and z axes, respectively; $v^b_{w,x}$, $v^b_{w,y}$, $v^b_{w,z}$ are the wind speed projected onto the body frame; ω_x, ω_y, ω_z are the angular velocity components about the body-fixed x, y, and z axes, respectively; K_{Dx}, K_{Dy}, K_{Dz} are the drag coefficients of the drone geometry that depends on air density and geometry; ϕ, θ, ψ are the roll, pith and yaw angles; m is the total mass of the quadrotor; I_{xx}, I_{yy}, I_{zz} are moments of inertia about the body-fixed x, y, and z axes, respectively; I_p is the propeller moment of inertia; $F_{T,i}$ is the thrust force of each motor with $i = 1, 2, 3, 4$; $\tau_{D,i}$ is the drag torque produced by each rotor; $\dot{\omega}_{p,i}$ is the rate of change of angular velocity of each rotor; a,b are the distance from the center of the quadrotor to the center of each rotor.

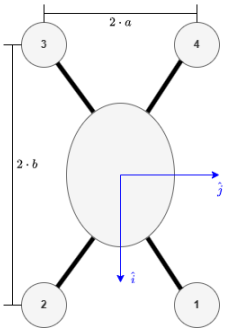
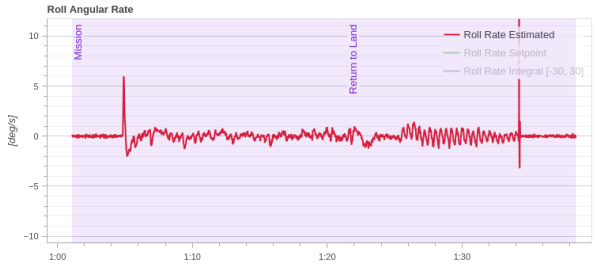
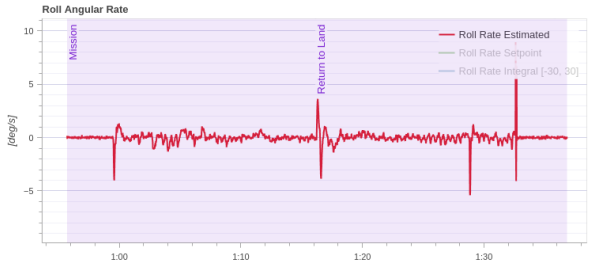


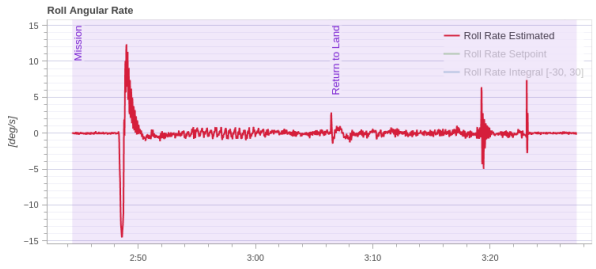
Fig. 15: Illustration of drone used for analytical derivation



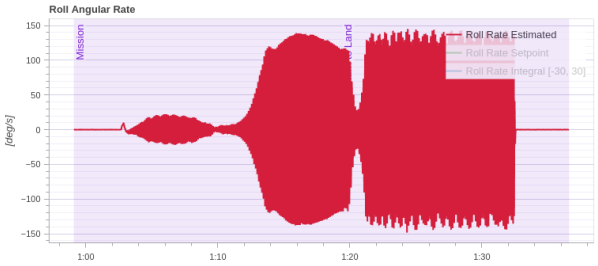
(a)



(b)



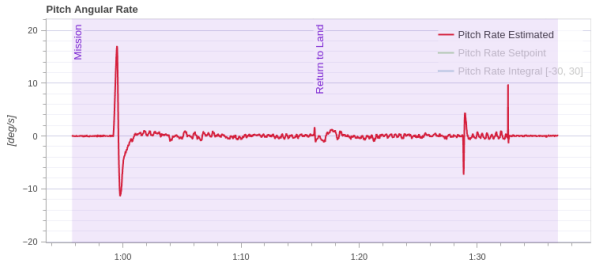
(c)



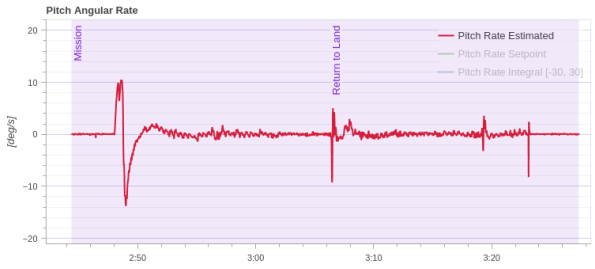
(d)

Fig. 16: Roll Angular rate for each simulation. Respectively a)drone with dimensions 0.275×0.1975[m] and null motor latency, b)drone with dimensions 0.275×0.1975[m] and motor latency equal to 6ms, c)drone with dimensions 0.55×0.395[m] and null motor latency, d)drone with dimensions 0.55×0.395[m] and motor latency equal to 6ms

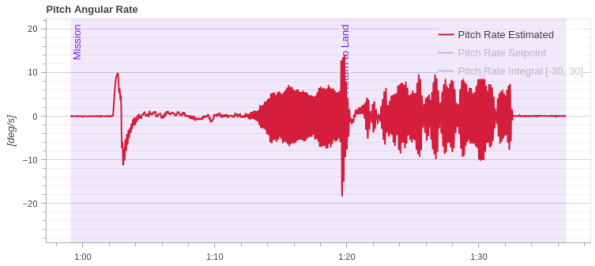
(a)



(b)

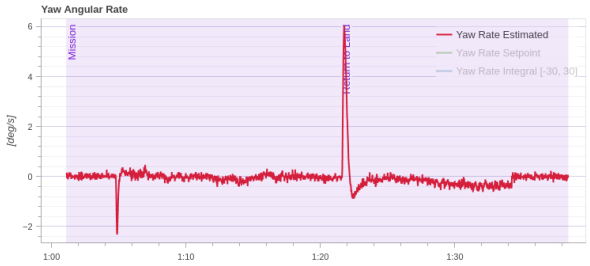


(c)

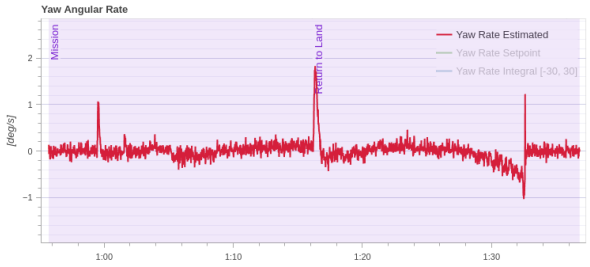


(d)

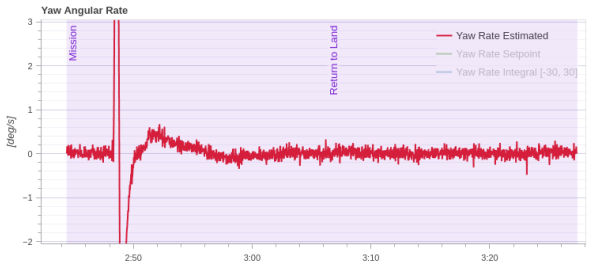
Fig. 17: Pitch Angular rate for each simulation. Respectively a)drone with dimensions 0.275×0.1975[m] and null motor latency, b)drone with dimensions 0.275×0.1975[m] and motor latency equal to 6ms, c)drone with dimensions 0.55×0.395[m] and null motor latency, d)drone with dimensions 0.55×0.395[m] and motor latency equal to 6ms



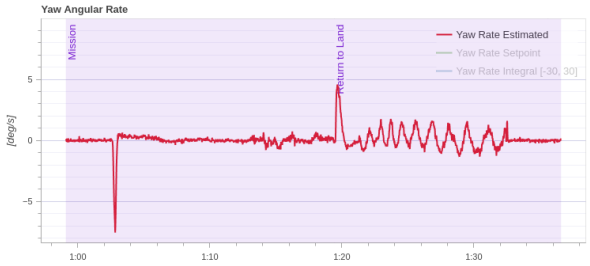
(a)



(b)

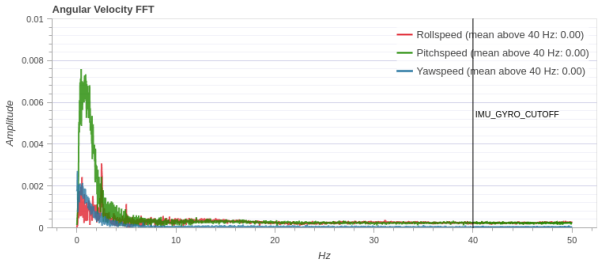


(c)

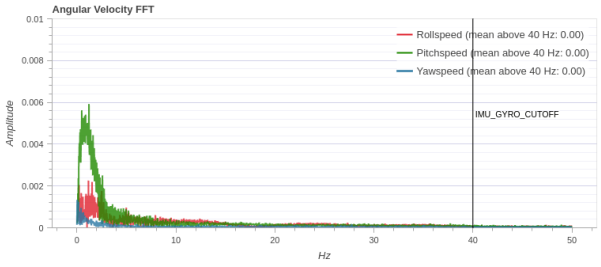


(d)

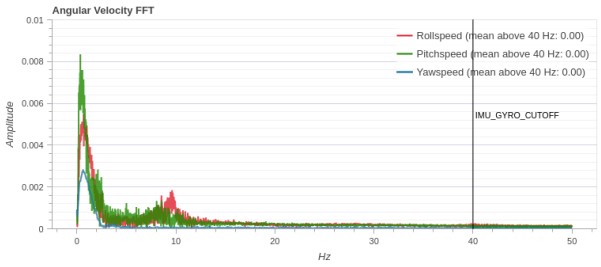
Fig. 18: Yaw Angular rate for each simulation. Respectively a)drone with dimensions 0.275×0.1975[m] and null motor latency, b)drone with dimensions 0.275×0.1975[m] and motor latency equal to 6ms, c)drone with dimensions 0.55×0.395[m] and null motor latency, d)drone with dimensions 0.55×0.395[m] and motor latency equal to 6ms



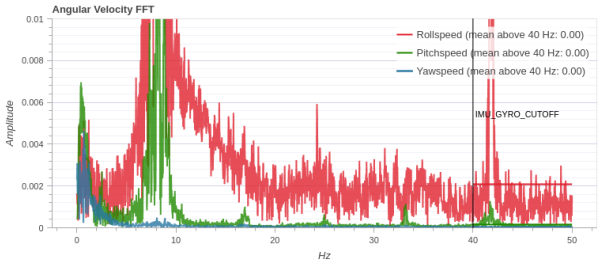
(a)



(b)



(c)



(d)

Fig. 19: Angular Velocity Fast Fourier Transformation rate for each simulation. Respectively a)drone with dimensions 0.275×0.1975[m] and null motor latency, b)drone with dimensions 0.275×0.1975[m] and motor latency equal to 6ms, c)drone with dimensions 0.55×0.395[m] and null motor latency, d)drone with dimensions 0.55×0.395[m] and motor latency equal to 6ms

V. Receiver Delay Improvement

A further aim of the project was to analyse and improve the communication between the master module and the slave module. A first step was to understand the cause of the delay previously mentioned and described in Figure 5.

The failure to update the duty cycle in the second period of the T-REC receiver module is due to the previous acquisition scheme adopted.

The receiver module uses the `nrf_drv_pwm` driver, which is part of the Nordic SDK and provides a higher-level API for

interacting with the PWM hardware. In fact, the Pulse Width Modulation (PWM) module driver comprises two levels: the Hardware Access Level (HAL) and the Driver Layer (DRV). The hardware access layer offers simple APIs to access the PWM peripheral registers. The Driver Layer provides higher-level APIs than the HAL. More information of how to implement a PWM signal using the driver library are available on the site of Nordic Semiconductor [8].

In order to use the PWM Driver Layer correctly, the following summarized steps must be followed:

1) First, call `nrf_drv_pwm_init` to initialize and configure the driver for a specific PWM instance. Which takes as input a pointer to the driver instance structure, a pointer to the structure with initial configuration (`nrf_drv_pwm_config_t` and event handler provided by the user. For example `nrf_drv_pwm_init(&m_pwm0, &config0, NULL);`
2) Next, define a sequence of duty cycle values using the `nrf_pwm_sequence_t` structure
3) Start the playback of the sequence by calling `nrf_drv_pwm_simple_playback`
4) If the whole sequence definition must be changed, use `nrf_drv_pwm_sequence_update`
5) At any time, the playback can be stopped by calling `nrf_drv_pwm_stop`

What has been noted in the previous work is that the full functionality and options of the PWM driver have not been exploited. In fact, when a new PWM value is received, it is not updated using the function `nrf_drv_pwm_sequence_update` and the internal compare registers are not updated in time before the start of the new period.

The improvements in the code consists in a direct manipulation of the registers of the `NRF_PWM0` module, which is a lower-level approach that gives more control over the hardware. Therefore, the Driver Layer has not be used. [9]

The new receiver code has been tested with the same experiment done in the previous work. Figure 20 shows one PWM channel of the FC with period *T-FC* and the replicated PWM on the receiver side with period *T-REC* and auxiliary signal (scaled down by a factor of ten) for debugging purposes.

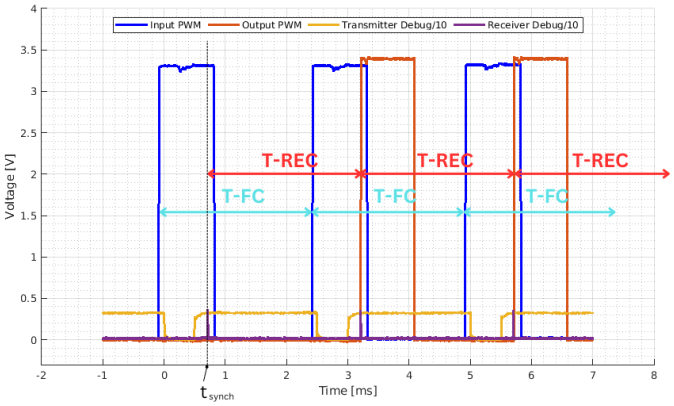


Fig. 20: Delay in the new reply scheme

As can be seen, the PWM signal at the input of the transmitter module is replicated by the receiver module with one period less than in the previous system. As a result, the new system has a total delay of approximately 3295 [us].

The simulation described in Section IV can be run again using in this case a drone geometry of 0.55×0.395[m] with a motor delay of 3.3[ms]. As can be seen from the analysis of the Fourier transform of the angular velocity shown in Figure 21, the drone vibrates significantly less.

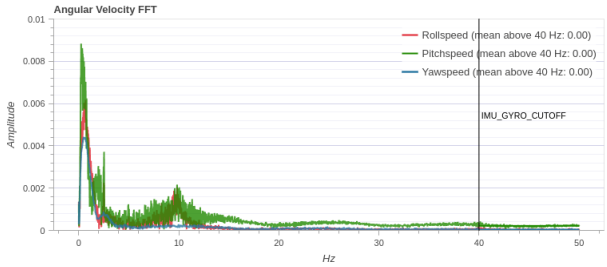


Fig. 21: Angular Velocity Fast Fourier Transformation of a drone with dimensions 0.55×0.395[m] and with a motor delay of 3.3[ms]

A. *Change ESC protocol considerations*

As shown in Figure 5 and as previously discussed, we could see that the time it hypothetically takes the master-slave system, via UWB, to communicate is approximately:

$$\begin{aligned} \Delta t &= (t_3 - t_2) + (t_4 - t_3) + (t_5 - t_4) \\ &= 72us + 509us + 189us \\ &= 770us \end{aligned} \tag{5}$$

Consequently, one could hypothetically have a PWM signal with a minimum period of $800[us]$, i.e. with a frequency of $f = 1.25[kHz]$ instead of $400[Hz]$. Unfortunately there are no protocols that meet these conditions. The closest would be the *OneShot125* with a signal width of $125 - 250us$ and therefore with a frequency of $4[kHz]$, and consequently too high for the new system implemented. These newer protocols (e.g. Oneshot125, Oneshot42, Multishot, DShot, ProShot) are similar to PWM, but they have much shorter signal width

leading to a lower latency. Dshot for example don't need ESC calibration as the protocol uses digital encoding.

VI. Flight tests

A. Drone Geometry & Calibration phase

Before proceeding with flight tests, it is appropriate to dwell on the calibration phase. Which includes setting up the drone's compass, gyroscope, accelerometer, and other sensors to ensure they are finely tuned to the drone's specific characteristics and operating environment. For the calibration phase, the steps described in the PX4 manual were followed, but in the case of the following project, a few things should be noted: the airframe geometry selection and the respective motors position with respect to the Flight Controller it's important because the physical configuration of the drone's frame affects its aerodynamics and flight characteristics. Selecting the correct airframe geometry ensures that the drone can handle the intended flight patterns and loads. The four receiver modules were attached to a cardboard box with dimensions $0.60 \times 0.40 \times 0.40$ [m] and weigth of 1.2 [kg]. In the flight test carried out, it was decided to adopt an H-geometry as the arrangement of the motors as shown in the Figure 22 recalls this configuration. The position of the motors is the same as that used in the simulation in Section IV e.g. $[(x_i, y_i)] = [(0.275, 0.1975), (-0.275, -0.1975), (0.275, -0.1975), (-0.275, 0.1975)]$ [m]. Since the drone's payload is approximately $0.645 * 4 + 1.2 = 3.78$ [Kg] the *Turnigy Multistar 4822 690KV* motor with $13\times5\times2$ propellers was used. With this configuration, the drone is able to lift a theoretical safety load of 7[kg].

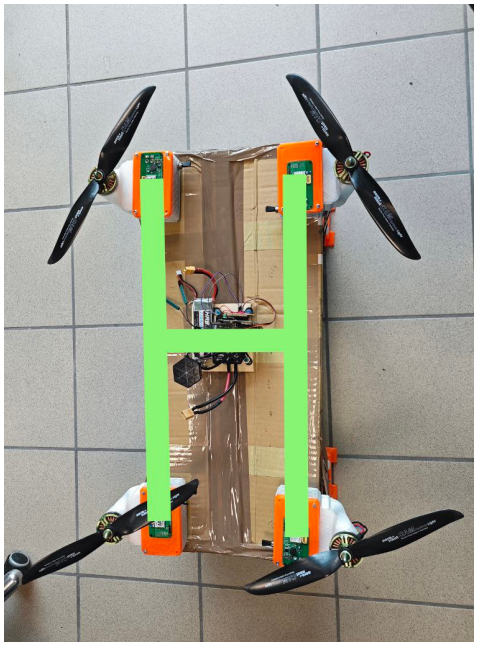


Fig. 22: H airframe selection

A further aspect to be emphasised is the ESC calibration phase as they commonly vary a lot in their response to input. ESC calibration updates the ESCs with the maximum and minimum PWM input values from the flight controller. This uniform response is crucial, especially when performing precise manoeuvres or carrying payloads.

A wired calibration is preferred over calibration via UWB. The latter is still possible but less accurate. It is therefore necessary to connect the ESCs to the Flight Controller via cable. In the specific case of this project, the steps to follow are:

1) Remove the propellers.
2) Map the ESCs you're calibrating as motors. Only mapped actuators get an output and only ESCs mapped as motors will be calibrated.
3) Connect the flight controller to the computer using a USB cable without connecting the battery to the Power Distribution Board (PDB).

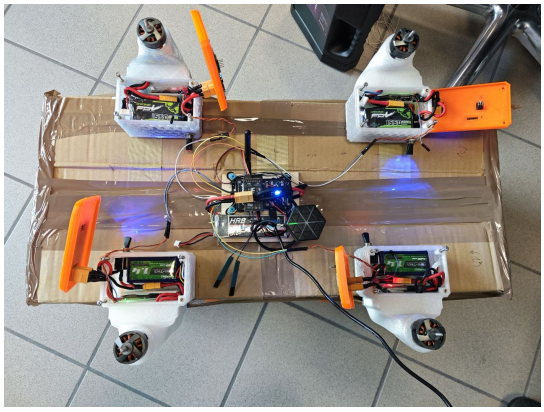


Fig. 23: Wired ESC Calibration phase

4) Open the QGroundControl Settings>Power, then press the Calibrate button. After starting the calibration sequence without error connect the battery to the PDB of the FC, while remaining connected to the PC via USB cable.
5) The calibration will begin automatically
6) Wait until the calibration completion prompt is shown and ESC has finished beeping.

Since PX4 has no feedback from the ESC to know whether or not calibration was successful you have to rely on interpreting the beeps during the calibration and subsequent motor tests to know for sure that the calibration worked.

B. Flight tests

Finally, two generic flight tests were performed using the drone-box with the H configuration. The difference between the two tests lies in the firmware used by the receiver modules in which the delay was reduced from 6[ms] to 3.3 [ms]. The purpose of these tests is to see how the delay affects using a certain geometry. What emerged from the tests is that in both configurations the drone flew correctly and from the analysis of the flight data there are no major differences in terms of vibration and stability. This is in fact in line with what was found in the simulations carried out in Section IV. The geometry of the drone is not large enough to find large differences in stability. What was instead noticed during the tests is a slight improvement in terms of the drone's

responsiveness to the control given by the radio control. The following figures show the data analysed using the Flight Review online tool and Matlab from the flight log test with the new firmware on the receiver modules.

Figure 24 shows the estimated trajectory of the drone during the flight test.

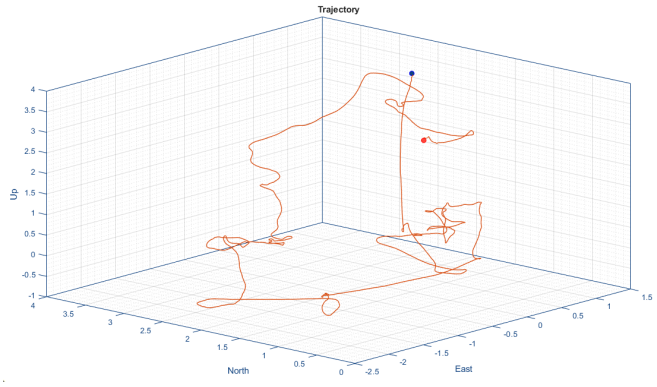
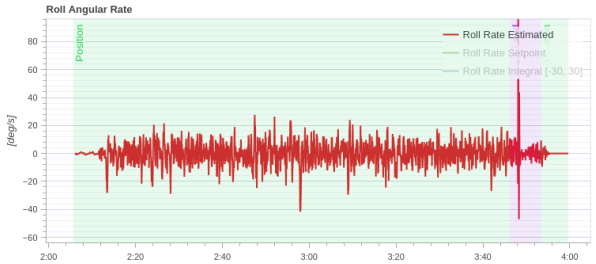
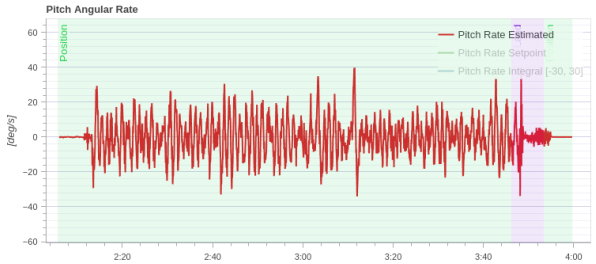


Fig. 24: Estimated drone trajectory

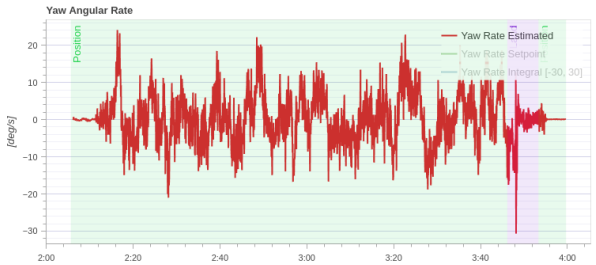
Overall, the results of the flight test mounting a custom PCB for the receiver module are very gratifying as the drone vibrated visually very little. As can be seen from the acceleration data shown in Figure 26 the z-axis touches the x/y-axis graph relatively little, which indicates as a general rule of thumb that vibration levels are low. These oscillations not only remain limited to amplitude values but remain constrained to relatively low frequencies as shown in Figure 27, where the fast Fourier transform of the angular velocity is calculated.



(a) Roll Angular Rate



(b) Pitch Angular Rate



(c) Yaw Angular Rate

Fig. 25: Angular Rates of drone-box flight test with H-motor configuration

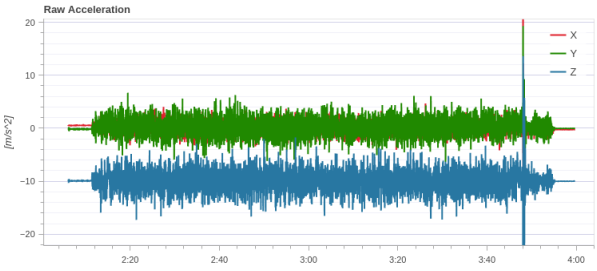


Fig. 26: Drone's raw Acceleration

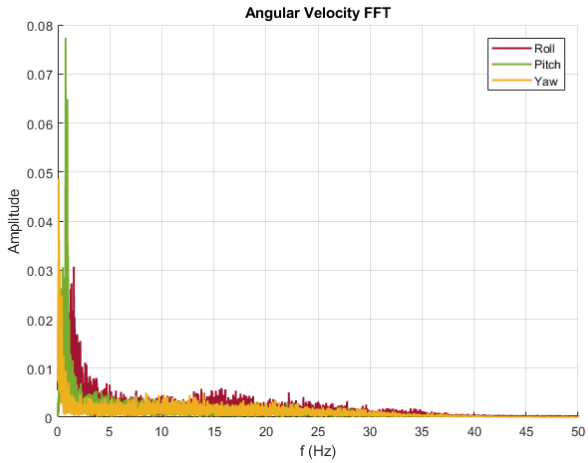


Fig. 27: Angular Velocity Fast Fourier Transform

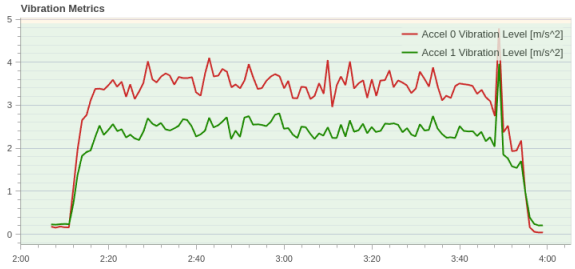


Fig. 28: Vibration level

The actuator outputs graph in Figure 29 shows the signals that are sent to the individual motors in the range between the minimum and maximum configured PWM values (e.g. from 1000us to 2000us). As can be seen the mean value, calculated over the entire flight period is close to the maximum value. In fact, the controller often goes into saturation. This is an indication that the vehicle is overweight for the amount of thrust that it can provide. This plot also serves as an indicator of the vehicle's overall balance. The motor with the smallest actuating output on average is the fourth with 1912 [us] and the motor with the highest actuating output is the first with 1966 [us]. It can also be noted that the output actuation of the third motor with 1956[us] while the second motor has 1937[us]. These results suggest that the box drone was not fully balanced and that the centre of mass was consequently shifted slightly forward. An imbalanced vehicle is generally not a big problem as the autopilot will automatically account for it. However it reduces the maximum achievable thrust and puts more strain on some motors. In our specific case where the four motors are powered separately, a greater imbalance would lead to a faster battery discharge of one module than the others.

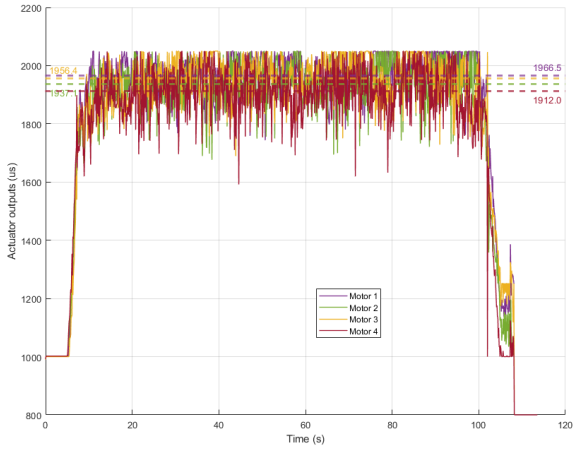


Fig. 29: Actuator Outputs

C. Receiver module power consumption

An important aspect when designing a drone is energy consumption as it affects flight duration. The primary factors to be taken into account are the payload, the components to be powered and the propulsion system. With regard to energy consumption, the solution adopted has been found to consume a similar amount of energy as other drones. In the previous work, the output power of the custom receiver PCB board at different voltages was tested and estimated to be approximately 0.55[W]. This result is consistent with the specifications of the UWB *DWM1001* module, which states that the chip, powered at 3.3[V], consumes an average of 132[mA] with peaks of 150[mA] during reception. Since the receiver module can be considered to be in permanent reception mode (as it receive signal at 400Hz), the power consumed is approximately 3.3[V]$\cdot 132$[mA]$=0.44$[W], a value that is comparable to the previous result and therefore in the order of a few watts. In contrast to the receiver module, the propulsion system requires power in the order of hundreds of watts. Considering the motor *Turnigy Multistar 4822* the datasheet indicates that powering it at 14.8[V] it operates at a current of less than 16[A] (with maximum peaks of 22[A]). This gives an average power of about 14.8[V]$\cdot 16$[A]$=237$[W]. As can be seen, we have that the highest power consumption of a module is due to the propulsion system, making the new UWB communication technology negligible in terms of energy consumed. We can therefore estimate the lifetime of each module (with this specific hardware configuration) to be approximately $\frac{1400[\text{mAh}]}{16[\text{A}]} \cdot 60 \approx 5.25[\text{min}]$.

VII. Conclusions and Future Work

The aim of this project was to continue the project previously started by other students so that the prototype of a modular drone using Ultra Wide Band technology to communicate could be improved in terms of hardware and software. After a detailed analysis of the components used and the work previously carried out, the receiver module software was improved in which Hardware Access Level (HAL) was used to control the PWM signal. The latter offers simple

APIs to access the registers of the PWM peripheral, a lower-level approach but offering more control over the hardware. This made it possible to decrease the delay by approximately one period from 6ms to 3.3ms. This delay was calculated as the time taken by the receiver module to replicate the signal generated by the flight controller connected to the transmitter module. SITL simulations were then carried out to see how varying the positioning of the motors and varying the delay changes the flight stability of the drone. The results showed that as the distance between the motors increases, greater flight instability is generated. A new slave module that could be attached to any object was therefore designed and printed in ABS using a 3D printer. This module proved to be an excellent compromise between ease of insertion (and removal) of components and space occupied. Finally, flight tests were carried out to validate how the new mechanical design mounting the selected hardware and customised PCB communication boards can be a major step forward in the development of efficient and reliable drones that have no geometric constraints.

The future goals and the next improvement could be:

- The introduction of algorithms to estimate the relative position of modules. This would allow for a plug-and-play drone where modules could be positioned in an arbitrary shape.
- Further improving the communication protocol by removing the PWM readout from the loop and using an FC that communicates digital information, e.g. via the CAN bus, with the transmitter module. In this way, the only delay left would be the one required for preparing, sending and receiving the message via UWB. The use of a CAN protocol is widely used in wired drones where ESCs are controlled via the DroneCAN software protocol. For this purpose, additional hardware must be used for the transmitter module to be able to receive CAN messages. Several methods are available, one of which would be to use a *CAN-to-UART converter* which converts messages from the CAN protocol to the UART protocol and thus readable by the transmitter module.
- Investigating how motor delay can be managed or predicted by appropriately modifying the flight controller.
- Design a support structure for the transmitter module that can include the FC, battery, GPS and radio frequency communication system so that it can be stably attached to any object.
- Further SITL or HITL analyses and simulations can be carried out by varying various parameters such as the geometry of the drone and consequently its inertia, the delay to the motors, the number of connected modules and how the latter are attached to a generic object, finally validating mathematically the results obtained. In particular, it would be useful to explore methods of identifying the UAV's dynamics through an identification procedure based on standard estimation tools.

REFERENCES

[1] 2006. URL https://www.itu.int/dms_pubrec/itu-r/rec/sm/R-REC-SM.1755-0-200605-I!!PDF-E.pdf.

[2] Rafael Vargas-Bernal. Introductory chapter: Novel developments in uwb technology. In Rafael Vargas-Bernal, editor, *UWB Technology*, chapter 1. IntechOpen, Rijeka, 2023. doi: 10.5772/intechopen.110609. URL https://doi.org/10.5772/intechopen.110609.

[3] Yusnita Rahayu, Tharek Abd. Rahman, Razali Ngah, and P.S. Hall. Ultra wideband technology and its applications. In *2008 5th IFIP International Conference on Wireless and Optical Communications Networks (WOCN '08)*, pages 1–5, 2008. doi: 10.1109/WOCN.2008.4542537.

[4] URL https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.nrf52832.ps.v1.1%2Fppi.html.

[5] Gabriel Staples. Propeller static & dynamic thrust calculation - part 2 of 2 - how did i come up with this equation?, Apr 2014. URL https://www.electricrcaircraftguy.com/2014/04/propeller-static-dynamic-thrust-equation-background.html.

[6] Oscar Liang. Oscarliangpropellers, May 2023. URL https://oscarliang.com/propellers/.

[7] Ultimate 3d printing material properties table. URL https://www.simplify3d.com/resources/materials-guide/properties-table/.

[8] Nordic Semiconductor. nrf52-pwm-driver, . URL https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk5.v12.0.0%2Flib_pwm.html.

[9] Nordic Semiconductor. nrf52-pwm, . URL https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.nrf52832.ps.v1.1%2Fpwm.html&anchor=concept_wxj_hnw_nr.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[11] Lauren Nagel. How to calculate & measure propeller thrust, Mar 2023. URL https://www.tytorobotics.com/blogs/articles/how-to-calculate-propeller-thrust.