Deconstructing for Superiority: Aristotelian Modularization in UAV Systems

Luca Santoro, Alessandro Assirelli, Erik Mischiatti, Marco Perini, Matteo Nardello,
Daniele Fontanelli and Davide Brunelli

Abstract— **In recent years, the widespread adoption of unmanned aerial vehicles (UAVs) has posed significant challenges in their design. Developing a UAV involves careful consideration of factors such as size, rotor count, propeller and motor dimensions, thrust requirements, and battery type. These specifications are selected based on the intended application, considering aspects like flight duration, payload capacity, maneuverability, and safety. As a result, there is an increasing demand for diverse UAV models tailored to meet specific application needs effectively. This paper presents a novel method for establishing communication between a flight controller and the electronic speed controllers in a UAV, utilizing ultra-wideband (UWB) technology. We present a solution for building a modular drone with UWB technology and an analysis of its communication system. Our results demonstrate the effectiveness of our approach, providing a practical solution for creating efficient UAVs without geometric constraints. This solution shows potential for applications in the challenging field of smart logistics.**

I. Introduction

Over the past decade, there has been a significant transformation in people's expectations regarding delivery services, transitioning from longer wait times to daily delivery options. Companies like Amazon have capitalized on this shift, making it a cornerstone of their business model. However, existing logistics infrastructure struggles to efficiently meet this demand. In response, there has been growing interest in drone delivery services, particularly since Amazon announced plans to utilize unmanned aerial vehicles (UAVs) for deliveries. Advancements in micro-electro-mechanical systems (MEMS), sensors, navigation techniques, remote control capabilities, and power storage systems have facilitated the development and production of a wide array of UAVs. These robots vary in size, configuration, and capabilities, enabling their use in diverse circumstances and tasks offering effective solutions to challenges such as high logistics costs and the delivery of goods orders in remote areas. In conclusion, the integration of UAVs into logistics operations is becoming increasingly prevalent, representing an inevitable trend for future development and research: UAV logistics is poised to become an integral part of people's daily lives.

As outlined above, UAVs are currently utilized across a wide range of applications, presenting a significant challenge in their design. When developing a UAV, different factors must be considered, including its size, the number of rotors, the dimensions of the propellers and motors, the required thrust, and the type of battery to be utilized. These specifications are chosen based on the intended application, taking into account factors such as flight duration, payload capacity, maneuverability, and safety requirements. Consequently, there is a growing need for a diverse range of UAV tailored to effectively and efficiently fulfill specific applications with suitable performance levels. As a consequence, the existing literature addressing topics such as drone design [1], theoretical frameworks [2], implementation strategies [3], [4], and experimental findings [5], explores a multitude of approaches and techniques for integrating modular or adjustable features into UAVs, enabling physical reconfiguration [6].

Studies comparing the performance of modular and traditional UAVs have suggested that embracing a modular approach may lead to time and energy savings during flight missions [7]. Various approaches are available in the current state-of-the-art literature for designing modular UAVs. They can be broadly categorized into three strategies. The first one makes the UAV easily reconfigurable, allowing for adjustments such as moving the rotor arms or changing their length [8]. For instance, DARPA's Morphing Aircraft Structures (MAS) program was developed to create shape-changing UAVs that integrate optimal performance capabilities into a single system[1]. In [9] a novel and straightforward morphing design for quadrotors was proposed. This design features a frame with four independently rotating arms that fold around the main frame. Additionally, in [10], a multirotor modular system kit was introduced, enabling the creation of a modular UAV with various configurations, which differ in terms of the number of arms, legs, rotors, motors, and landing capabilities.

Another approach entails combining small UAVs that can operate independently or as a cohesive unit to construct complex structures on the fly as required. For example, in [11], ModQuad-Vi was introduced as a novel modular quadrotor capable of docking with other modules of the same type using visual servoing. Lastly, there's the modular drone approach, wherein various modules are assembled to create a functional flying drone. This concept is illustrated by the Distributed Flight Array [12]. This vehicle comprises autonomous single-rotor modules capable of driving, docking with each other, and flying in synchronized formation. These modules function as distributed computational units with limited sensory input.

This paper proposes a solution that adopts a similar concept to the third approach outlined earlier, thus utilizing

Department of Industrial Engineering
University of Trento, Trento, Italy
name.surname@unitn.it

[1]`https://www.aerosociety.com/news/transforming-flight/`

Manuscript 650 submitted to 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE). Received March 25, 2024.

a central body for attaching various actuators. However, in the proposed scenario, the main body can vary, serving either as a custom airframe or as part of the payload, depending on the specific application. Another module houses the onboard computer responsible for controlling autonomous flight and stabilizing the UAV by transmitting commands to the actuators. This setup relies on wireless communication between the master module and the actuators, referred to as slaves in the proposed architecture. While previous research has explored the feasibility of wirelessly controlling the speed of DC motors using PWM techniques [13], [14], these studies primarily focused on the viability of the control approach without evaluating its performance in practical applications. Our work not only follows this wireless control approach but also assesses its performance on the drone through experimental analysis and validation. Technically this paper contributes to the existing body of literature results in different aspects, such as:

- The development of a custom UAV to implement the proposed architecture;
- The development of the firmware for signal acquisition and transmission via UWB communication;
- Development of custom Gazebo plugin to simulate delayed motor control chain.

The rest of the paper is organized as follows: Section II outlines the methodology for acquiring the control signal and transmitting it via ultrawide-band, examining the individual components of the proposed architecture. Section III details the hardware utilized in the experiments, presenting analyses from both hardware-in-the-loop simulation and real-world experiments, finally, Section IV concludes the paper with final remarks and suggestions for potential improvements.

II. Architectural Design

To design a versatile system capable of fulfilling diverse requirements, a modular architecture allows for seamless adjustments to its features, like the number of rotors or the airframe's structure. The proposed modular design also comprises separate modules that can be attached to any object to function as the UAV airframe.

A. Communication technology

Ultrawide-band (UWB) technology represents a wireless communication innovation leveraging nanosecond, low-power radio pulses to transmit high-bandwidth data such as video and audio wirelessly [15] across numerous devices. UWB is used in a wide range of applications including, indoor positioning [16], robot localization [17], radar imaging [18], and wireless sensor networks [19]. The large transmission bandwidth of UWB technology grants robustness against interference and multipath effects. This leads to heightened reliability and diminished susceptibility to signal degradation.

The hierarchical structure of the paper proposed solution, shown in Figure 1, adopts a master module communicating with multiple slave modules. Rather than directly transmitting signals to the Electronic Speed Controllers (ESCs),

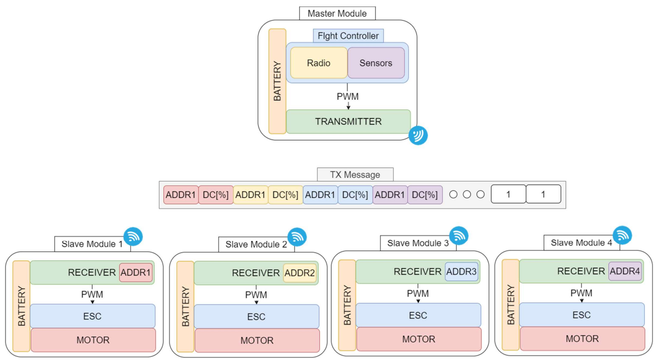

Fig. 1. Architectural design proposed

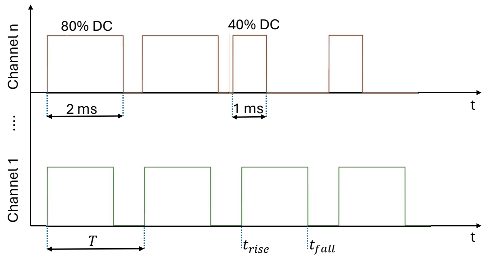

Fig. 2. Quadcopter FC PWMs example

the microcontroller unit (MCU) of the master module first collects duty cycle (DC) information. Then, the transmitted data is received by the receiver modules, which then establish the pulse-width modulation (PWM) duty cycle for the ESC connected to the motor.

B. Architecture

The architecture revolves around the master and slave modules. The **master module** is composed of three fundamental components powered by a LiPo battery, each playing a pivotal role in orchestrating the system's operations and ensuring seamless communication and control:

1) **Flight Controller**: Leveraging data from onboard sensors like accelerometers, gyroscopes, and magnetometers, this component generates control signals for the motors to stabilize the aircraft by generating PWM signals. In the case of a quadcopter, it generates four PWM signals. These PWM signals have a fixed frequency of $f_{FC} = 400$ Hz and are synchronized, ensuring that the rising edges of every channel occur simultaneously. The falling edges are determined by the DC of the specific channel. The minimum DC, corresponding to 0 rpm, is set at 40% (equivalent to a minimum duration of 1 ms), while the maximum is 80% (equivalent to a maximum duration of 2 ms), representing full throttle, as illustrated in Figure 2.
2) **UAV Radio Receiver**: A radio receiver works as a device capable of capturing commands transmitted by the radio transmitter. These signals are interpreted through the flight controller, which translates them into specific actions to effectively manage and control the aircraft.
3) **Acquisition Module**: Its primary role is to capture PWM signals from the Flight Controller (FC) and relay

duty cycle information to the receiver modules via UWB signals.

The Programmable Peripheral Interconnect (PPI) system facilitates PWM acquisition. PPI enables peripherals to communicate independently, utilizing tasks and events without CPU intervention. This feature ensures precise synchronization between peripherals, which is especially crucial in real-time scenarios, and minimizes the need for CPU involvement in executing predefined behaviors via PPI. The synchronization clock of PPI operates at a frequency of 16 MHz.

Each PWM channel is assigned a dedicated timer, guaranteeing that the timer counter is promptly and automatically stored in the register upon detecting a rising or falling edge. To retrieve the stored information representing the duty cycle, it is essential to differentiate between the times of the rising and falling edges. This is achieved by saving the counter value in the register when an edge is detected, without the CPU's intervention. Subsequently, the CPU enters an interrupt routine specifically designed to discern the edge type and store the counter value in memory. The duty cycle of the signals is DC $= \frac{t_{fall} - t_{rise}}{T}$, where t_{fall} and t_{rise} represent the times when the falling or rising edges occur, respectively, and T denotes the period of the signal, which is set to 2.5 ms in the proposed configuration. To transmit the PWM information to the receivers, the address of each receiver followed by the corresponding duty cycle is encoded into the message payload. This message is then transmitted at the end of each input PWM period.

The **Slave Module** can be integrated in any desired quantity to enhance the maximum payload capacity or extend flight distance, while adhering to common architecture standards such as those for tricopters, quadcopters, hexacopters, and beyond. Each slave module comprises a LiPo battery, an ESC, a motor with a propeller and a UWB receiver module. For the slave module, it's imperative to replicate the PWM signals generated by the FC. This is achieved by connecting the GPIO output of the slave module MCU to its timer, which is configured to produce a square wave with a constant frequency of $f_{SM} = f_{FC} = 400$ Hz. The timer initiates upon receiving an initialization signal from the master module, dubbed as a sync signal. Every $T = 2.5$ ms (i.e., PWM period generated by the FC), each receiver module decodes messages from the master module. It then identifies its address and extracts the duty cycle needed to transmit to the ESC for PWM signal generation (refer to Figure 1 for an illustration of the message structure).

III. Results

The aim of this paper is to thoroughly examine the viability of the proposed design by evaluating the delay injected by the communication system and its subsequent effects on UAV performance. As our baseline, we have utilized a commercial-off-the-shelf Pixhawk flight controller renowned for its reliable performance. Our attention will be dedicated to analyzing deviations from this baseline performance attributable to the communication system.

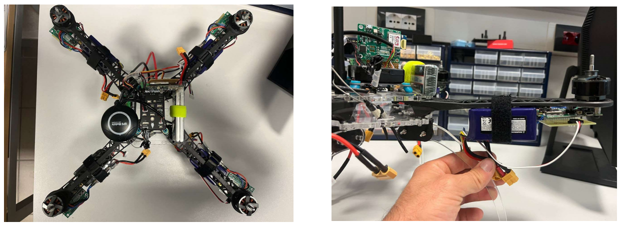
(a) (b)

Fig. 3. In (a) top view of the prototype; in (b) a slave module.

To validate the efficacy of the proposed solution, we performed two experiments. The first experiment aimed to determine the maximum number of slave modules that could be accommodated, thus establishing the maximum number of PWM signals that could be acquired. This assessment considered both the resources available in the chosen MCU and the employed acquisition technique. The second experiment focused on measuring the overall delay introduced by the proposed architecture within the motor control chain. These measurements were conducted using a PicoScope 5444D MSO oscilloscope [2].

A. Hardware

The master and slave modules are built upon the commercially available Decawave DWM1001C System on Module (SoM). This SoM incorporates a low-power nRF5283 MCU, the Decawave DW1000 UWB transceiver[3] and the LIS2DH12, an ultra-low-power high-performance three-axis linear accelerometer[4].

Throughout the experiments, the UWB modules were configured to operate on UWB Channel 5, utilizing a centered frequency of $f_c = 6489.6$ MHz, a bandwidth of $BW = 499.2$ MHz with a preamble length of 128 symbols, a pulse repetition frequency of $PRF = 64$ MHz, and data rate of $DR = 6.8$ Mbps.

The architecture is implemented on a customized version of a commercial quadcopter airframe (see Figure 3) class 500 mm, crafted from transparent laser-cut 5 mm plexiglass to showcase the absence of wiring between the flight controller and the motors. We chose this setup to contrast the performance between an optimized architecture and the proposed solution, thereby highlighting any deviations in expected behavior. In total, the system utilizes five LiPo batteries: one for the master module and one for each slave module.

B. Maximum Number of Slave Modules

As previously explained, when a GPIO event occurs (i.e., rising or falling edge of the PWM signal), the timer counter is promptly stored in the register. This process occurs nearly instantaneously, given that the PPI clock operates at a frequency of 16 MHz.

[2]https://www.picotech.com/download/datasheets/picoscope-5000d-series-data-sheet.pdf
[3]https://www.qorvo.com/products/p/DW1000
[4]https://www.st.com/en/mems-and-sensors/lis2dh12.html

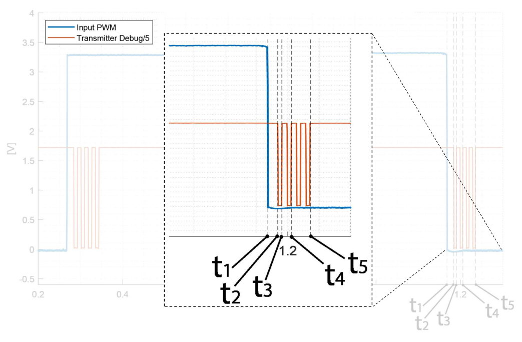

Fig. 4. The time required to store the counter register of each channel in memory. The red trace indicates the state of a debug pin on the DWM1001. It is set to a low logical state when the CPU enters the corresponding Interrupt Service Routine (ISR) and returns to a high state once the channel reading is completed.

The register needs to be read and stored in memory before the opposite edge occurs because it gets overwritten in both the rising and falling cases. The PWM generated by the flight controller has a duty cycle ranging from a minimum of 40% to a maximum of 80% of T. To ensure a reliable signals acquisition procedure, each register of every channel acquisition must be read within a 20% window of T, corresponding to $t_{acq}^{(max)} = 500\ \mu$s.

To measure the time required to store the counter register of each channel in memory, we conducted an experiment using the master module to read four PWM signals from the flight controller. Leveraging a debug pin on the DWM1001, its logical state is set to low when the CPU enters the corresponding interrupt service routine (ISR) and to high when the reading of the channel is complete. Notice that the measurement will also include the time necessary to toggle the pin, a factor deemed negligible in comparison to the overall quantity being measured. Figure 4 presents the findings of this experiment. Only one input PWM signal is depicted for the sake of clarity, as the other three are synchronised. The time required to store the counter in memory consists of two components. The first component is the time taken to enter the ISR, computed as $\Delta_{21} = t_2 - t_1$. The second component is the time needed to identify which channel is generating the interrupt and copy the counter value into memory, which is $\Delta_{32} = t_3 - t_2$. The time interval Δ_{32} remains constant regardless of the number of channels used, whereas the time required for the CPU to initiate the execution of ISRs varies. Specifically, when the signals are synchronised, such as in the flight controller scenario, the CPU will simultaneously receive multiple interrupts. This results in the generation of an interrupt queue because only one ISR can be executed at a time. When there is a queue of ISRs, executing the first ISR (referred to as Δ_{21}) typically requires more time compared to the subsequent ones due to the overhead involved in context switching. Context switching involves saving the current state of the processor, including the program counter and register values, and loading the state required for the next ISR. This overhead is incurred each time an ISR is serviced, which means that the first ISR in the queue experiences the full overhead of

TABLE I
DELAYS SUMMARY OF FIGURE 4.ALL QUANTITIES ARE EXPRESSED IN μS.

Δ_{ij}	Value	Description
$\Delta_{21} = t_2 \quad t_1$	16	ISR entry time
$\Delta_{32} = t_3 \quad t_2$	7	Channel and Edge Detection to Register Storage Time
$\Delta_{31} = t_3 \quad t_1$	23	Total Time for Storing Register Value of First Channel
$\Delta_{43} = t_4 \quad t_3$	15	Total Time for Storing Register Value of Additional Channels
$\Delta_{51} = t_5 \quad t_1$	68	Total Time to Store Four Channels

TABLE II
DELAYS SUMMARY OF FIGURE 5. ALL QUANTITIES ARE EXPRESSED IN μS.

Δ'_{ij}	Value	Description
$\Delta'_{32} = t'_3 \quad t'_2$	72	Communication start delay
$\Delta'_{43} = t'_4 \quad t'_3$	509	Message encoding and transmission
$\Delta'_{54} = t'_5 \quad t'_4$	189	Message reception , decoding a parsing payload
$\Delta'_{65} = t'_6 \quad t'_5$	59	Time margin on the next pwm period
$\Delta'_{71} = t'_7 \quad t'_1$	5831	Total injected delay

context switching, while subsequent ISRs may benefit from some optimizations or efficiencies introduced by the context switching mechanism.

In our experimental setup, storing all register values in memory requires approximately $\Delta_{31} = 23\ \mu$s for the first channel, while $\Delta_{43} = 15\ \mu$s for each additional signal acquired. Detailed delay data is provided in Table I.

Constrained by $t_{acq}^{(max)}$ to prevent the loss of any input signal edges, we calculate the maximum number of slave modules using Equation (1).

$$\text{max n. channels} = \frac{t_{acq}^{(max)} - \Delta_{31}}{\Delta_{43}} + 1 \sim 31, \qquad (1)$$

which is more than enough for standard applications.

To store the counter values in memory for four channels, the total time required is calculated as $\Delta_{31} + n\Delta_{43} = 68\ \mu$s, where n represents the number of additional channels to be acquired, which in the proposed architecture is $n = 3$.

C. Injected Delay in Motor Control Chain

Another crucial aspect is the introduced delay in the motor control chain, which must be measured from the time instant the FC sends a signal to when it is mirrored on the slave module side. This measurement excludes the ESC processing time, which is a constant factor present in both traditional and proposed setups. To assess this, the transmitter reads the four channels of the FC, transmits them, and subsequently replicates them on the slave module side.

The injected delay comprises various contributions, as outlined in Table II. Figure 5 illustrates a single PWM channel acquisition from the FC in blue, the replicated PWM signal on the slave module in red, and an auxiliary signal used for debugging purposes. Notice that the debugging signal is scaled down by a factor of ten for improved plot clarity.

Figure 5 displays a single PWM channel acquisition from the FC in blue, along with the replicated PWM signal on the slave module shown in red. The figure also includes auxiliary

Manuscript 650 submitted to 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE). Received March 25, 2024.

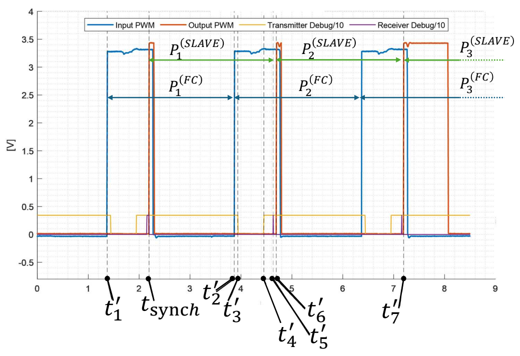

Fig. 5. Delays in the mirroring scheme. Injected delay components detailed in Table II. Blue trace: FC PWM acquisition. Red trace: Replicated PWM on slave module. Auxiliary debugging signals included, scaled down by a factor of ten for plot clarity.

signals used for debugging purposes. It's important to note that the debugging signal is scaled down by a factor of ten to improve plot clarity, as it shares the same output voltage level as PWM signals. The acquisition scheme employed on the master module, responsible for collecting duty cycle information, is required to wait until the PWM signal generation on the FC module completes, a process taking one period T. However, before transmission commences, an additional delay Δ'_{32} arises, during which the MCU must read and store timer registers in memory.

To quantify this delay, we initialize the debug pin on the master module to a low logic state at the onset of message encoding. Following this, the CPU constructs the message, appending the receiver addresses and duty cycle information to the message payload, before transmitting it to the slave modules. After the transmission is completed, the debug pin is set to a high logic state. The duration of these operations is denoted as Δ'_{43}.

The message arrives at the receiver with a delay that depends on the distance from the transmitter. However, this delay can be considered negligible compared to the other contributions. In this case the delay is collected by rising the value of a pin on the receiver side when the reception starts and by lowering it when it ends. The time required to receive the message and parse it is Δ'_{54}.

In Figure 5, note that the generation of the PWM signal on the slave module begins at t_{synch}, corresponding to a synchronization message sent by the master module. Consequently, the PWM signal initiates from a distinct arbitrary time instant compared to that produced by the FC.

At the rising edge of the second period $P_2^{(FC)}$, occurring at t'_2, the transmitter sends a message containing information about $DC_1^{(FC)} = \frac{t_{fall}^{(1)} - t_{rise}^{(1)}}{T}$, the duty cycle of the first period ($P_1^{(FC)}$) of the FC . Despite the message being received and parsed before the slave module initiates the second period $P_2^{(SLAVE)}$, the value of $DC_2^{(SLAVE)}$ remains unchanged.

To observe the duty cycle $DC_1^{(FC)}$ mirrored on the slave module, an additional period is required, resulting in the replication of the FC PWM at the period $P_3^{(SLAVE)}$. This delay stems from the hardware and software implementation

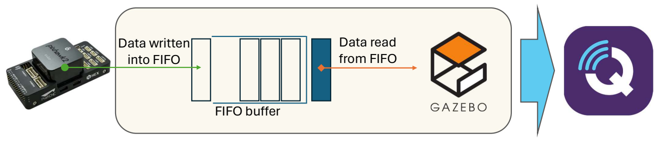

Fig. 6. Diagram depicting the HITL simulation stack incorporating a FIFO buffer. Output data flows through QGroundControl, acting as the intermediary between the simulated environment and the physical hardware under test.

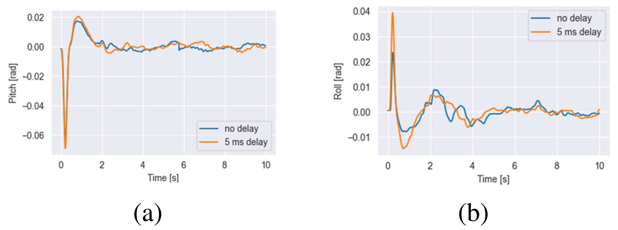

Fig. 7. Results of HITL simulation by integrating FIFO buffer. In (a) the results for the pitch angle; in (b) the results for the roll angle. Trace in blue refer to an injected delay of 0 ms, the orange trace refer to an injected delay of 5 ms.

of the Nordic library for PWM signal generation. Typically, if a sequence of duty cicles values is played in a loop and values need updating before the next iteration, modifications occur when the corresponding event signaling the end of the sequence. However, at this stage of operation, we exceed this critical timing, introducing an additional delay to retrieve data from the RAM and update values for PWM signal generation [5].

D. Simulation results

Hardware-in-the-loop (HITL) simulation is a valuable method for testing and validating flight control software in a virtual computer environment, replicating real-world systems' behavior. In HITL, sensor readings are generated by the simulated environment, while calculations to control the drone are performed using a real flight controller connected through serial communication.

To simulate the UWB connection between the flight controller and ESCs, a Gazebo plugin to introduce a configurable delay in the drone PWM transmission is developed. The delay is modeled as a First-In-First-Out (FIFO) buffer (refer to Figure 6) to simulate the latency injected by the proposed system and assess the performance of the flight control software. By modifying the delay within the Gazebo plugin, we could assess the FC's capability to stabilize and control the drone across two latency scenarios. Figure 7 illustrates pitch and roll angles values with latencies of both 0 and 5 ms. It showcases that the flight controller stack maintains stable flight performance even with significant delays in the control motor chain. As a result, we have concluded that a latency of 5 ms should not compromise the stability of the UAV.

[5] `https://shorturl.at/dfNX6`

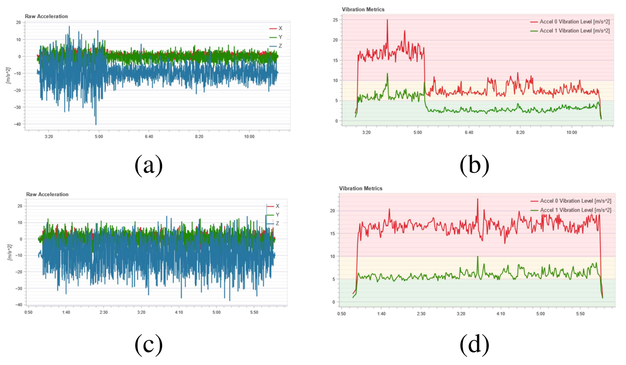

Fig. 8. In (a) and (c) UAV raw accelerations in traditional and wireless setup respectively, in (b) and (d) the vibration metric computed for the traditional and wireless setup respectively.

E. Real flight test

Following the simulation results, we conducted an analysis of the proposed architecture in real-world conditions. Our assessment focused on evaluating the performance by examining the raw accelerations in the UAV reference frame along three directions. Specifically, we experimentally evaluated the occurrence of excessive vibration levels during hovering maneuvers by observing if the acceleration along the z-axis intersects with the xy-plane. Vibration presents a common challenge for multirotor vehicles, manifesting in various adverse effects such as reduced flight efficiency, shorter flight durations, motor overheating, accelerated material wear, difficulty in achieving precise vehicle tuning leading to degraded flight performance, sensor clipping, and position estimation failures, which can result in fly-aways.

In the proposed scenario, this phenomenon is initially observed with the traditional setup, but it becomes more pronounced in the wireless setup, where vibration levels persist at high levels throughout the flight. However, despite this, the flight stack successfully stabilises the flight, aligning with the expectations from the HITL simulations. This confirms the validity of the proposed setup, as demonstrated in Figure 8, which displays raw accelerations and vibration metrics calculated as the norm of two consecutive acceleration values. The observed instability could stem from communication delays between the flight controller and the ESC module or potential asynchronous clocks among the slave modules. Both factors have the potential to introduce instabilities, leading to more significant oscillations in the actuators' input signals.

IV. Conclusion

In this study, we successfully designed and tested a modular drone prototype, showcasing the feasibility of creating a wireless motor control chain for UAV applications. Our design offers versatility and customization options, enabling easy adaptation to diverse applications and environments.

The proposed design presents a promising solution for more efficient and adaptable UAVs. Future endeavors will focus on refining communication systems between modules to enhance stability, conducting tests in challenging environments, and exploring potential applications for the modular design.

While this study demonstrates the modification of an existing UAV to validate the proposed concept, the next phase will involve developing customized wireless slave modules designed to be attached to objects and transport them.

References

[1] E. Gökbel, A. Güllü, and S. Ersoy, "Improvement of uav: design and implementation on launchability," *Aircraft Engineering and Aerospace Technology*, vol. 95, no. 5, pp. 734–740, 2023.

[2] D. Cawthorne and A. Robbins-van Wynsberghe, "An ethical framework for the design, development, implementation, and assessment of drones used in public healthcare," *Science and Engineering Ethics*, vol. 26, pp. 2867–2891, 2020.

[3] B. Sah, R. Gupta, and D. Bani-Hani, "Analysis of barriers to implement drone logistics," *International Journal of Logistics Research and Applications*, vol. 24, no. 6, pp. 531–550, 2021.

[4] R. P. Bremner and K. M. Eisenhardt, "Organizing form, experimentation, and performance: Innovation in the nascent civilian drone industry," *Organization Science*, vol. 33, no. 4, pp. 1645–1674, 2022.

[5] M. A. da Silva Ferreira, M. F. T. Begazo, G. C. Lopes, A. F. de Oliveira, E. L. Colombini, and A. da Silva Simões, "Drone reconfigurable architecture (dra): A multipurpose modular architecture for unmanned aerial vehicles (uavs)," *Journal of Intelligent & Robotic Systems*, vol. 99, no. 3-4, pp. 517–534, 2020.

[6] F. Schiano, P. M. Kornatowski, L. Cencetti, and D. Floreano, "Reconfigurable drone system for transportation of parcels with variable mass and size," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 150–12 157, 2022.

[7] J. Lee, "Optimization of a modular drone delivery system," in *2017 Annual IEEE Int. Systems Conference (SysCon)*, 2017, pp. 1–8.

[8] M. Erbil, S. Prior, M. Karamanoglu, S. Odedra, C. Barlow, and D. Lewis, "Reconfigurable unmanned aerial vehicles," *International Conference on Manufacturing and Engineering Systems. Proceedings*, pp. 392–396, Dec. 2009.

[9] D. Falanga, K. Kleber, S. Mintchev, D. Floreano, and D. Scaramuzza, "The foldable drone: A morphing quadrotor that can squeeze and fly," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 209–216, 2019.

[10] S. Brischetto, A. Ciano, and C. Ferro, "A multipurpose modular drone with adjustable arms produced via the fdm additive manufacturing process," *Curved and Layered Structures*, vol. 3, 07 2016.

[11] G. Li, B. Gabrich, D. Saldaña, J. Das, V. Kumar, and M. Yim, "Modquad-vi: A vision-based self-assembling modular quadrotor," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 346–352.

[12] R. Oung and R. D'Andrea, "The distributed flight array," *Mechatronics*, vol. 21, no. 6, pp. 908–917, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741581000139X

[13] A. Menaka, R. Seetharaman, and K. Anandan, "Wireless stepper motor control using rf-communication," in *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, vol. 2, 2022, pp. 640–645.

[14] J. M. Hinsu, P. B. Zala, V. V. Pandit, V. G. Vadher, and S. P. Khunt, "Wireless dc motor speed and direction control using rf communication," *International Journal of Novel Research and Development*, vol. 2, no. 4, 2017.

[15] W. Cui, P. Ranta, T. A. Brown, and C. Reed, "Wireless video streaming over uwb," in *2007 IEEE International Conference on Ultra-Wideband*, 2007, pp. 933–936.

[16] L. Santoro, M. Nardello, D. Brunelli, and D. Fontanelli, "UWB-based Indoor Positioning System with Infinite Scalability," *IEEE Trans. on Instrumentation and Measurement*, vol. 72, pp. 1–11, 2023.

[17] L. Santoro, D. Brunelli, and D. Fontanelli, "On-line optimal ranging sensor deployment for robotic exploration," *IEEE Sensors Journal*, vol. 22, no. 6, Mar. 2022.

[18] L. Santoro, M. Nardello, D. Fontanelli, and D. Brunelli, "UWB bistatic radar sensor: across channels evaluation," *IEEE Sensors Letters*, vol. 7, no. 10, pp. 1–4, Oct. 2023.

[19] Y. V. Lakshmi, P. Singh, S. Mahajan, A. Nayyar, and M. Abouhawwash, "Accurate range-free localization with hybrid dv-hop algorithms based on pso for uwb wireless sensor networks," *Arabian Journal for Science and Engineering*, vol. 49, no. 3, pp. 4157–4178, 2024.