A Distributed Integrated Modular Avionics platform for Multi-Mission Vehicles

Miguel Tavares de Barros
miguel.barros@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Portugal

July 2019

Abstract

More than ever, the development of avionics systems for unmanned multi-mission vehicles requires an ever-increasing optimization and standardization effort. Recently, a new type of avionics architecture based on the use of Distributed Integrated Modular Avionics (DIMA) was developed for civil aviation. It allows for a significant increase in the number of avionics functionalities and a reduction in related weight and power consumption. By adopting these DIMA architectures, unmanned multi-mission vehicles gain increased flexibility, certifiability and compatibility with regulated airspace, expanding their range of applications in a safe manner. This thesis details the existing limitations with unmanned platforms and proposes the use of DIMA architectures as a solution. To this purpose, an existing reconfigurable DIMA architecture was studied and integrated in an Unmanned Aerial Vehicle (UAV). The resulting platform was then validated in a simulated environment and tested in a real use-case scenario. Concurrently the use of Software-Defined Radio (SDR) communication for data-link applications was investigated, studying the benefits of integrating this flexible technology in such a modular avionics platform. The proposed avionics integration proved to be successful, in both the simulated and the real environments, performing an automated flight while maintaining its existing DIMA reconfiguration capabilities. A small data link protocol was also implemented using SDR, which worked as expected on the work bench, but struggled in larger ranges. It is concluded that the integration of DIMA architectures on multi-mission vehicles is not only viable, but encouraged, since these can attend to some current and relevant limitations of this industry.

Keywords: Integrated Modular Avionics, Multi-Mission Vehicle, Software-Defined Radio

1. Introduction

Over the past decades, the development of Multi-Mission Vehicles (MMVs) has increased substantially, taking advantage of their high degree of versatility and the multiplicity of their use-cases, and establishing this field as a critical market for civil and military industries. With the innovations in computing and automation, Unmanned Vehicles (UVs) have proved to be effective in a variety of fields from underwater and ground domains, to the space exploration industry. Originally, Unmanned Aerial Vehicles (UAVs) were mainly military aircraft, used in missions that were deemed too dull or dangerous for human pilots. With the rapid growth in the availability of technology across the world, new applications for these systems were found. Nowadays, UAVs are part of multiple fields of our society, as, for example, agricultural, scientific and recreational. Due to this quick rise in popularity and the variety of use-cases, a standardized avionics architecture for UAVs does not exist. Every manufacturer implements their own *ad hoc* solution, preventing the use of these vehicles as an abstract platform. With the ever-increasing integration of these vehicles in regulated airspace, concerns about the safety of these systems are being raised, leading to newer and more strict regulations. As such, the current approach of independent and unique avionics platforms, without any overarching architecture, may soon move to be outdated and obsolete.

In contrast, the civil air transport industry has converged into a common architecture since the 90s, based on Integrated Modular Avionics (IMA) solutions. These avionics platforms are the accepted solution for dealing with the pressure for increasing aircraft functionality while keeping avionics systems safe. Distributed Integrated Modular Avionics (DIMA), also known as Second Generation IMA (IMA2G), is a recent and promising concept in IMA, which further improves modularity, reliability and lowers the cost when comparing with older architectures [6]. This kind of proven innovative avionics platform can be the solution for future

UAV integrations, solving concerns of reliability, flexibility and safety.

This work studies and furthers the development of an existing DIMA prototype UAV platform, tailoring it to fly in an aerial multi-mission vehicle. At the same time, this work also studies a radio technology named Software-Defined Radio (SDR). SDR technologies offer a promising novel approach to communication, permitting a higher degree of flexibility and reconfigurability over typical communication infrastructures. Hence, SDR may prove to have extensive synergies with IMA architectures and its inclusion in an aircraft could have several benefits for Size, Weight and Power (SWaP) requirements.

2. Background

In order to provide the proper background, we will start by detailing the limitations of current UAV avionics platforms and describing how the use of Integrated Modular Avionics can provide significant improvements to the industry.

2.1. The current state of UVs

Today, the use of unmanned vehicles in our society is ever-increasing, ranging from recreational to commercial and scientific uses. The recent growth in this industry is being lead by multiple factors such as: continuous developments in associated technologies, an increase and diversification of applications and the widespread use of location based services. However, there are some challenges that these systems still face. If solved, these would allow unmanned vehicles to become a more popular, safe and connected part of our lives.

Cost and reliability — While the price of unmanned systems and equipment is low when compared to the commercial aviation industry, cheap avionics components still have low technological maturity, reliability and a higher than desired failure rate. A cost-effective increase in reliability would further the use of this kind of platforms.

Safety — The number of UAVs using shared airspace has been growing significantly and there are still few enforceable certification requirements that would guarantee safety of these systems. At the same time, most control and safety requirements for these platforms are still too relaxed, leading to less need for strict real-time requirements. The introduction of a partitioned architecture for these platforms would ease the certification efforts, providing a cost-effective path for complying with future safety regulations.

Flexibility — The large number of possible missions a UAV can perform implies that any underlying platform must be capable of supporting the highly variable requirements needed. Different missions can require distinct payloads, from high storage and bandwidth cameras, to strict real-time sensors. The flexibility of the platform is limited to the supported payloads. For this reason, the creation of a standard interface that would allow the introduction of new payloads to the system would greatly benefit the flexibility of the system.

Automation and high-performance computing — Currently, most of the data obtained in a mission follows one of two paths. The live transmission of the raw data to the ground station, or the storage of the data for posterior analysis. Both these solutions can be troublesome and could be improved by expanding the on-board processing capabilities of the aircraft. This would greatly reduce the amount of data to be transmitted, which would allow the operator to quickly evaluate if mission objectives are being fulfilled and, if necessary, intervene in real-time.

2.2. Federated architectures and IMA

Until the early 1990s, civil aircraft avionics systems followed a federated architecture. The cornerstone of this architecture is to use a segregated “black box”, called a Line-Replaceable Unit (LRU), for each avionics function. All the hardware, software, and Input/Output (I/O) interfaces needed by the system are self-contained in LRUs. Due to the modular nature of this architecture, some benefits arise when designing such systems. These modular elements are easy to replace, allowing for simple fault detection mechanisms and maintenance, since one only needs to swap out the affected box for a similar one in stock. On the other hand, the inner design of the LRUs is usually developed by a third party contractor, preventing the final system integrator to know specific details about the equipment, increasing the cost of modifications and upgrades to the system.

Contrary to the federated architectures, IMA architectures focus on providing a shared pool of computing, communication and I/O resources to different systems. As such, spare capacity can be allocated where needed, and the number of different hardware part types is reduced. This new approach proved to be a huge success, resulting not only in a state-of-the-art avionics system, as well as in a new model for developing future ones [8]. Although computing resources may be shared between hosted functions, each one may have specific interface needs. In the first IMA implementations, the processing modules were developed with the I/O interfaces needed by all the applications using that resource. This forced each function to one or more predetermined physical units, since those were the only ones with the specific I/O interface. These processing modules were regularly placed in the aircraft’s avionics bay and had to be connected through these specific I/O interfaces, to the respective sensors and actuators. This required extra cabling and, while still an advance over traditional federated architectures, left room for improvement. IMA platforms then evolved into a new paradigm where the I/O tasks are decentralized, extending the use of the Remote Data Concentrator (RDC) modules and placing them much closer to the sensors and actuators of the aircraft, reducing even more the amount of cabling required. Additionally, by focusing the processing modules solely on hosting the avionics functions, this enabled the use of general purpose Central Processing Units (CPUs), decreasing the heterogeneity of the platform and consequently, its cost. Nowadays, research in IMA technologies continues to push the boundaries in these distributed system by integrating new and important concepts, such as incremental certification and reconfiguration mechanisms.

2.3. IMA architectures and its benefits

As was explored previously, one core benefit of IMA solutions is the optimization of resource allocation, minimizing the existence of unused computing capacity in the system and improving SWaP requirements. Additionally, IMA platforms have also shown relevant enhancement in safety parameters as Mean Time Between Failures (MTBF). Aamir Mairaj studies the comparison between federated and IMA architectures in several projects [3]. In Figure 1, one can observe and compare the impact caused by the introduction of this technology.

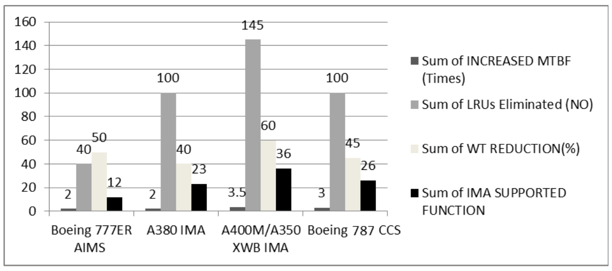

Figure 1: Comparison of IMA architecture impact in different platforms [3].

All the studied platforms improved their MTBF between 2 and 3.5 times, proving that, even without physical segregation between systems, IMA platforms can lead to better reliability. Additionally, the use of IMA eliminated the need for 40 to 145 LRUs, providing a weight reduction of 40% to 60%. Of the four presented projects, the Airbus A350 is the one showing greater improvements, which is expected since it makes most use of this different paradigm. IMA platforms also bring a different kind of advantage due to their specific development process. With the introduction of IMA, the integrator takes a more active role in sub-system development. With the abstraction layer created by the Real-Time Operating System (RTOS), software developed by the third-party companies is also more portable. At the same time, as developers of avionics functions are no longer in charge of hardware necessities, their focus can be shifted onto hosted software, easing development burdens and certification efforts [7].

2.4. DIMA as a platform solution

As one can see, distributed IMA opens the path for a more connected, open and flexible architecture. At the same time, it does not compromise safety, allowing for the certification of a platform to highest levels of assurance. This study will focus on the impact of DIMA solutions applied to UAV avionics platforms, where concepts which were previously only applied to non-critical systems, such as "Plug-and-Play" and reconfiguration mechanisms, could later be safely introduced.

2.5. Software-Defined Radio solutions

In order to further study the possibilities of a DIMA architecture for UVs, this work will conduct a study on the feasibility of integrating Software-Defined Radio (SDR) solutions as an air-to-ground data link. In software-defined radio communication, components that are commonly implemented in hardware, such as modulators and demodulators, are instead implemented in software. This enables higher flexibility as it is easier to change the waveform or the encoding when these elements are defined by software. In addition, SDR tentatively offers some advantages over hardware based solutions, such as the possibility of reconfiguring links in real-time, better management of communication spectrum by cooperative selection of channels and more aggressive error-checking techniques.

These advantages make SDR a good candidate for integration in a reconfigurable IMA network such as DIMA. The configurability of DIMA can be expanded to encompass also the radio frequency communication link. A usage example of such integration is the selection of the wireless communication link properties depending on the requirements of a given payload. Such a functionality also maps well with different mission phases, where the same transmitter can adapt to the availability of different communication links.

3. The first DIMA platform

The main motivation in the development of the initial DIMA platform was to expand IMA to the market of unmanned vehicles, addressing some trends in this industry. Most UAVs are still based on proprietary and closed architectures, where a handful of suppliers have full control of sub-system integration. At the same time, certification concerns for these systems are increasing, as a result of their expansion in applicability and desire to operate in regulated airspace.

3.1. The distributed IMA architecture

To answer these goals, the platform utilizes a distributed IMA architecture as its foundation. The use of IMA standards also introduces an underlying partitioned architecture, where the avionics platform is composed of different layers, which abstract the underlying hardware and allow applications to be developed independently. By combining the versatility of UAVs with the open modularity and proven safety of DIMA technologies, the goal is to introduce the concept of "UAV Platforms as a Service", while improving the current standards for safety and security in UAVs.

3.2. Reconfiguration mechanisms

One key enabler for this vision is the introduction of reconfiguration mechanisms into distributed IMA platforms, which plays a critical role in opening new possibilities. There are various benefits to using reconfiguration in IMA, some of which are detailed below.

Overcoming hardware failure — The platform can continuously verify each module's state and take action if one fails. Reconfiguration allows it to overcome hardware failure by using redundancy management, improving the operational reliability of the aircraft while preserving current levels of safety.

Multiple levels of autonomy — At the same time, reconfiguration allows the unmanned vehicle to work in different autonomy levels. The user can be in charge of flying the aircraft, or leave that to the autopilot and dedicate its efforts somewhere else, for example navigation or payload deployment. This allows vehicles to be optionally piloted, or to work with reduced crew configurations.

Change in Payload/Mission — Current avionics system designs are mission specific, and switching between them is time and cost consuming. Reconfiguration allows the platform to react and adapt to a payload or mission change, saving important resources between different missions. Payload integration can be handled by software, excusing the need for dedicated hardware. By using reconfiguration mechanisms the platform can change to meet the different payload's requirements.

In DIMA, the implemented reconfiguration mechanism is classified as automatic and multi-static. Automatic reconfiguration occurs without human involvement, after being triggered by a predetermined event. Multi-static reconfiguration states that the set of configurations and transitions between configurations are predefined and prevalidated. This prior verification of allowed states opens the way for a certification process, as their expected behavior has been previously studied. The main trigger for the reconfiguration in DIMA is a modification in the payloads attached to the avionics system. After user consent is given, the platform exercises reconfiguration at all levels of the system, changing the applications running on each module, adapting network routes and providing applications with new parameters (e.g. new aerodynamic variables or increased weight).

3.3. DIMA Test Bench

To prototype the creation of such avionics platform and the study of reconfiguration solutions, a DIMA Test Bench was developed and is represented in Figure 2. The DIMA Test Bench includes four Core Processing Modules (CPMs) and four RDCs, emulated using inexpensive boards. These are connected by an Ethernet network which allows them to exchange useful information.

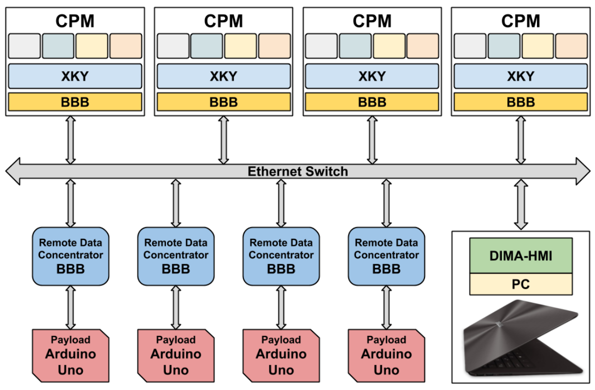

Figure 2: DIMA Test Bench block diagram.

The Test Bench is a small scale demonstrator, and, given the prohibitive overhead of integrating real payloads in the system, these are simulated using Arduino prototyping boards and other quick prototyping elements. The core processing modules used run GMV's real-time operating system, XKY [1]. These elements are responsible for the main processing of the avionics platform as well as the reconfiguration services. The remote data concentrators run RTEMS real-time operating system. Their function is to act as a bridge between the Ethernet network and the payloads of the system. They also implement the payload recognition protocol and participate in the system-wide reconfiguration. More detail on this platform and on the reasoning behind its development choices can be found in *Distributed Integrated Modular Avionics* [2].

3.4. Outcomes

The DIMA Test Bench was a successful prototype of a distributed IMA platform, allowing the exploration of new technologies and reconfiguration mechanisms. While its objectives were fulfilled, the platform was still far from being representative of a real-case scenario. With the promising results obtained, this work continues the research by further developing DIMA and studying some other interesting new concepts related to IMA technologies.

4. A DIMA platform for UVs

Having described the existing platform, we now present the avionics platform for a DIMA based UAV, aptly named DIMA-FLY. The steps towards its development are hereby discussed, mainly involving the integration of the Commercial Off-the-Shelf (COTS) flight control system and the prototyping of the SDR data link. The main objectives of the study were:

1. To integrate a tailored DIMA platform in a UAV, providing automated flight functionality and payload driven reconfiguration;

2. To study and prototype the usage of SDR in DIMA architectures.

All these needed to be fulfilled without compromising the reconfiguration functionalities featured in the original DIMA Test Bench.

4.1. UAV acquisition and assembly

The aircraft that was integrated with DIMA avionics is Applied Aeronautics' Albatross UAV. The Albatross UAV is a high performance commercial drone with an affordable price, which uses composite materials, weighs around 4kg (without batteries), has a 10kg Maximum Takeoff Weight (MTOW) and a 3m wingspan.

4.2. Integration of the Flight Control System

One of the main goals for the DIMA-FLY platform is the capability of automated flight. The problem of autonomy in UAVs has been studied heavily and there are now commercial off-the-shelf solutions that could be integrated in the avionics platform. Using a COTS module prevents the need to develop solutions for problems outside of the scope of the study, such as sensor device drivers and a fail-safe manual mode. The work involved choosing the proper hardware and software, studying their implementations and designing a way for the DIMA platform to cooperate with the Flight Control System (FCS), providing some flight autonomy. The chosen FCS was the Pixhawk 1 hardware board, using the PX4 software stack.

4.3. Operating flight modes

The ability of a system to control the movement of a vehicle can be divided into three main tasks: Guidance, Navigation and Control. To explore different levels of responsibility for the DIMA system while making most of the FCS functionalities, two flight modes were designed:

Waypoint mode — The simpler mode, where the DIMA platform is only in charge of the Navigation, and delegates Guidance and Control tasks to the FCS. The DIMA platform is in charge of receiving the desired trajectory waypoints from the ground station and communicating them to the FCS, while safely executing other previously defined tasks as payload handling and health-monitoring.

Attitude-target mode — This mode uses DIMA to execute both Navigation and Guidance tasks, leaving only lower level Control tasks to the FCS. Contrary to the previous mode, this one requires low latency processing to provide the desired instantaneous attitude to the FCS, furthering the test of this avionics platform as a potential solution for more representative use-cases.

4.4. Communication

The DIMA platform uses a regular Ethernet connection and UDP/IP to communicate between devices and the application layer of this communication uses a protocol named MAVLink. MAVLink (Micro Air Vehicle Link) is a protocol designed for communication between a ground station and unmanned vehicles. This protocol has become a standard in commercial off-the-shelf flight control systems, allowing for an agnostic approach to the Guidance and Navigation problem. As with our implementation, MAVLink can also be used as a communication protocol between on-board computers.

4.5. DIMA module application

The application which runs on the the DIMA core processing modules was written in ARINC 653 compliant C code. It was aptly named Guidance, Navigation and Control (GNC) and its responsibility is to handle requests from the ground station, use MAVLink to communicate with the Pixhawk and execute the desired calculations. The GNC application handles the bridge between the Ground Control Station (GCS) and the FCS. Depending on the mode of flight, this application has different functionalities. In the DIMA Waypoint mode, the pilot provides waypoints for the aircraft through the GCS, which are directed to the GNC partition. From there, the GNC partition handles the Navigation aspect, translating the waypoint directions to the MAVLink protocol and communicating them to the Pixhawk, which is then in charge of doing Guidance and Control tasks. In the DIMA Attitude-target mode, the pilot also provides waypoints through the GCS. In this mode however, the GNC partition handles the Guidance aspect, along with the Navigation, calculating the necessary position and attitude setpoints to accomplish the mission. The position and attitude target data are then sent obeying the MAVLink protocol to the Pixhawk, which handles only the Control aspect of finding the necessary control surface movements needed. The system is duplicated in a cold redundancy scheme. If one of the duplicated applications stops emitting a heartbeat message, the backup module takes responsibility for the system functionality using DIMA's reconfiguration mechanisms to immediately take charge.

4.6. Guidance algorithm

For the DIMA Attitude-target mode, the existing Guidance algorithm from the PX4 flight control software was adapted and implemented to run in DIMA processing modules. It uses two main calculations for lateral and longitudinal control of the aircraft.

4.6.1. Lateral control

The lateral control calculation is based on a nonlinear guidance algorithm, which is known as L_1 guidance algorithm. As described by Park et al. [4], this nonlinear algorithm starts by selecting a reference point in the desired path, which should be located at guidance length L_1 in front of the aircraft. Afterwards, a lateral acceleration command is determined using the expression:

$$a_s = 2\frac{V^2}{L_1}\sin\eta \tag{1}$$

Where a_s is the lateral acceleration of the aircraft, V the aircraft velocity, L_1 is the line from the aircraft position to the reference point and η is the angle from V to the line L_1 (clockwise is positive).

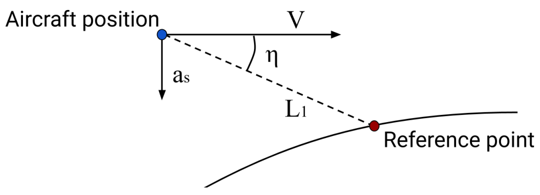

Figure 3: Guidance algorithm diagram.

This simpler approach to the lateral guidance problem provides two important properties:

1. The direction of the acceleration depends on the sign of the angle between the L_1 line and the aircraft velocity, allowing the vehicle to align the direction of its velocity with the one of the L_1 line.

2. At any point in time a circular path can be defined by two points (the position of the reference point and the current aircraft position) and a tangent (the aircraft velocity). This allows the algorithm to follow a circle when producing acceleration a_s.

The last step involves calculating the desired roll angle from the obtained acceleration. For this, the aircraft is described as being in a banked turn, where altitude is maintained and the centripetal acceleration is provided by the lift force and the roll angle.

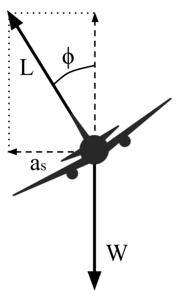

Figure 4: Banked turn diagram.

Using this model, the following equations are derived:

$$L\sin\phi = ma_s \tag{2}$$

$$L\cos\phi = W = mg \tag{3}$$

Where L is the aircraft lift, ϕ is the roll angle, W is the aircraft weight, m is the aircraft mass and g is the acceleration of gravity. Consequently:

$$mg\tan\phi = ma_s \tag{4}$$

$$\phi = \arctan\frac{a_s}{g} = \arctan\frac{2V^2\sin\eta}{gL_1} \tag{5}$$

The desired roll angle is then ready to be sent to the PX4 software, which will use it as a desired setpoint. The yaw angle is simply sent as the bearing from the current position to the L_1 reference point.

4.6.2. Longitudinal control

The longitudinal control is used to calculate the desired pitch and normalized throttle, allowing the aircraft to climb and descend automatically. For this, an approach called Total Energy Control System (TECS) is implemented in PX4. This main concept of this algorithm is described by R. Bruce [5]. It aims to control the total energy rate of the aircraft, which can be expressed as the sum of the potential and kinetic energy of the system:

$$E = Wh + \frac{1W}{2g}V^2 \tag{6}$$

Where E is the total energy of the aircraft and h is the aircraft altitude. The specific energy rate of the system, normalized for the velocity, can then be expressed by:

$$\frac{\dot{E}}{V} = \frac{\dot{h}}{V} + \frac{\dot{V}}{g} = \gamma + \frac{\dot{V}}{g} \tag{7}$$

Where γ is the flight path angle. For an accelerated climb, the following equation of motion is true:

$$T - D - W\sin\gamma = \frac{W}{g}a \tag{8}$$

From Equations 8 and 7, and for a small enough γ, we then obtain the thrust required as:

$$T = W(\gamma + \frac{\dot{V}}{g}) + D \tag{9}$$

Where D is the aircraft drag and T the required thrust. Assuming that the variation in drag is slow enough, Equations 7 and 9 allow us to approximate that the aircraft throttle controls the rate at which energy can be added or removed from the system. A control law can then be implemented, driving the error of the total energy rate to zero. Our implementation uses a simplified version of this control law, by porting the algorithm present in PX4 into the DIMA module. Utilizing a previously calculated desired setpoint for speed and altitude, we first obtain the demanded specific energy and specific energy rate. Using the cruise throttle as the starting point, we assume that:

1. The maximum specific total energy rate is achieved when throttle is at maximum;
2. A zero specific total energy rate is achieved when throttle is at cruise;
3. The minimum specific total energy rate is achieved when throttle is at minimum.

With this simplification, and utilizing the same Proportional–Integral–Derivative (PID) control feedback present in the PX4 software, we can calculate an appropriate normalized throttle that will match our desired specific energy rate. Upon obtaining this value, the final step is to specify a specific energy balance that details whether energy should be allocated towards kinetic or potential energy. Using calculations provided in the original implementation, this specific energy balance and the desired specific energy rate can be used to calculate a desired pitch angle. Together with the desired throttle, roll and yaw, these values are combined and sent to the PX4 software as a desired attitude setpoint.

4.7. Software-Defined Radio

Another goal of this study was to learn about and prototype a SDR data link. For this study a single channel data link was implemented. The two transceivers establish a communication channel. Upon arrival the message is subject to an integrity check, in order to guarantee the validity of the data. The radio communication operates in half-duplex mode avoiding cross feedback from the receiving and transmitting antennas. The SDR transceivers appear as static IP addresses in the DIMA network. As such, any packets that are to be sent through the SDR data link need only to be addressed at specific IP address of the RDC. This process provides a seamless integration of the SDR system, greatly simplifying its interaction with the network.

The SDR transceiver functionality, can be divided into two separate layers. The Application Layer, mainly responsible for receiving and sending data on the DIMA network, and the Physical Layer, responsible for transmitting and receiving frames of data between the application layer and the antennas, while executing needed signal processing functionalities. The SDR transceivers present in the system need to be physically connected to a host board or computer, which run the Application Layer software, responsible for several tasks in the data link communication channel. This software uses an Ethernet interface to communicate with the network. In the DIMA Ethernet network, the host boards are non-reconfigurable RDC modules with static IP addresses, bridging the gap between the SDR transceiver and the network. In the other side of the data link channel (the GCS), the SDR transceivers are connected to the computer performing ground control functions. Using preconfigured IP addresses and UDP ports the Application Layer listens to data and sends it across the channel. On the other end of the channel the other Application Layer instance forwards the data to the respective destination on the new network, connecting multiple receivers and sender across different networks. Despite the fact that this implementation uses only one computer on the GCS network, it still uses the configuration possibilities to send and receive different data through different UDP ports, allowing information exchange with different partitions in different CPM modules in the DIMA platform.

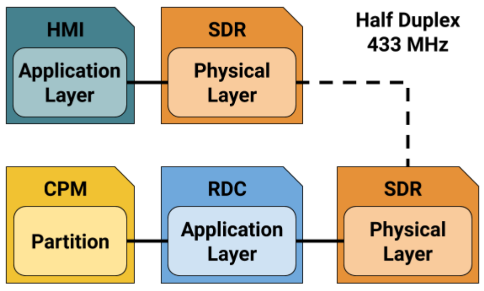

Figure 5: DIMA-FLY SDR functionality.

4.8. DIMA-FLY

After verifying the integrity of the Albatross airframe and testing the Pixhawk avionics, these were integrated into the Albatross aircraft. The Pixhawk board uses a variety of sensors and actuators to control the aircraft attitude according to the desired setpoint. The integrated sensors were:

- The internal Pixhawk sensors, as gyroscope, accelerometer, magnetometer and barometer;

- A pitot tube, fixed under the wing to measure static and dynamic pressure;
- And an external Global Positioning System (GPS) receiver, module Reyax RY836AI.

The integrated actuators were:

- Servomotors in control surfaces,
- Brushless motor for propulsion.

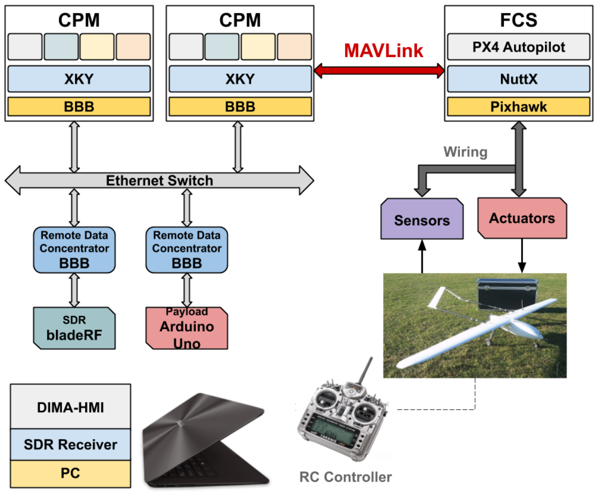

Figure 6: DIMA-FLY diagram.

5. Results

5.1. Software-In-The-Loop

The first validation of the integration was done through the development of a Software-In-The-Loop (SITL) configuration, testing the automated flight capabilities of the system in a simulated environment. A desktop PC running Linux was connected to the DIMA network. Here, two relevant programs were running: an instance of the previously mentioned PX4 software and a physics simulator named Gazebo.

5.2. Mode comparison

At this stage, it is important to study the comparison between the DIMA Waypoint mode and DIMA Attitude-target mode. The difference between them is that in DIMA Waypoint the Guidance task is up to the commercial FCS, while in DIMA Attitude-target mode the Guidance algorithm is calculated in the DIMA distributed modules. Using the same trajectory, data from both types of flights was obtained. Figure 7 shows the ideal trajectory, next to the ones resulting from Waypoint and Attitude-target modes.

As can be seen, the trajectory obtained by our implementation of the Guidance algorithm is similar to the one from PX4. In fact, the Attitude-target mode resulted in an Root Mean Square (RMS) lateral deviation of 3.984 m, while the Waypoint mode resulted in a higher 6.234 m deviation.

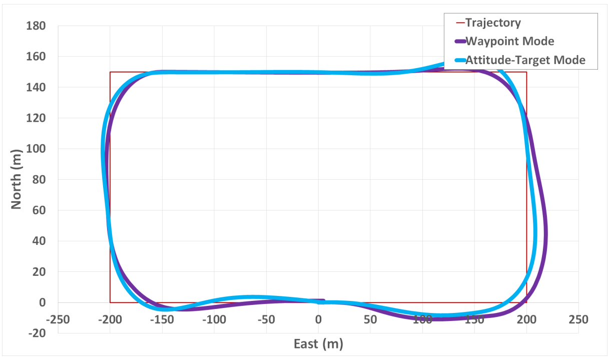

Figure 7: Trajectory comparison.

While this value is specific to this path, it serves as partial validation for the implemented algorithm, proving it can follow the desired lateral trajectory within a reasonable degree of accuracy.

The longitudinal component however, proved to be harder to replicate, as can be seen in Figure 8. Here, when ignoring takeoff and landing, Attitude-target mode resulted in an RMS vertical deviation of 5.263 m. Waypoint mode, on the other hand, resulted in a lower 3.940 m deviation. The PX4 implementation includes multiple adjustments to provide a more stable altitude which were not ported to the DIMA modules. While this does not completely invalidate the performance of the Attitude-target mode, it puts into question how robust it is and if it could handle unpredictable conditions as wind.

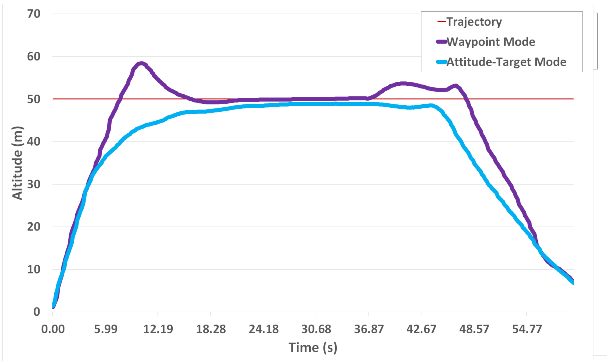

Figure 8: Trajectory altitude comparison.

5.3. Hardware-In-The-Loop

The next validation step was to repeat this simulation using a Hardware-In-The-Loop (HITL) mechanism. This means that the PX4 no longer runs on the Linux desktop, but now runs on the final hardware, the Pixhawk board. This allows testing the non-functional requirements of the system, guaranteeing the Pixhawk board has the needed memory and processing power to complete its tasks. The results of this step were similar to the one before, verifying that the system followed the expected behavior observed in the SITL simulation.

5.4. Aircraft-In-The-Loop

Finally, the DIMA-FLY was assembled in order to test the DIMA platform in a proper flight test. The purpose of this flight was to safely test the DIMA Waypoint mode, verifying the automation of the aircraft and the ability of the platform to handle other important tasks, such as payload handling and system-wide health-monitoring. The test flight started by using the PX4 Manual mode for takeoff. Afterwards, Stabilized mode was used to make sure that all the connections and signals are properly adjusted from the Pixhawk's perspective. When it was deemed safe, the DIMA GCS was used from the ground laptop, transmitting the desired waypoints to the flight controller. Then, DIMA Waypoint mode was initiated remotely, starting the automated portion of the flight. Using the Pixhawk / PX4 system allows the capture of important flight data in the form of a specific log file. Below, some relevant flight data that was retrieved from the test flight is presented.

5.4.1. Altitude estimate

As can be seen below, according to the GPS data, the ground altitude is at around 50 meters. The aircraft climbed in Manual mode until around 200 meters, when the Stabilized mode was activated. When the expected behavior of Stabilized mode was confirmed, Mission mode was activated from the Ground Control Station, and the aircraft flew its mission automatically at around 150 meters.

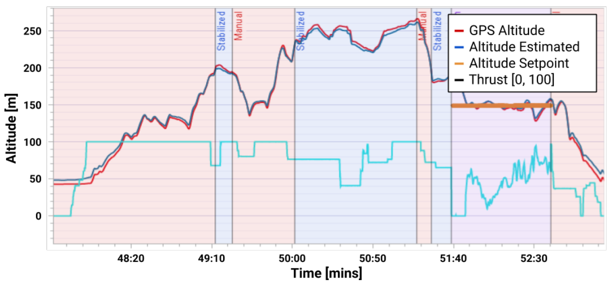

Figure 9: Altitude estimate.

5.4.2. Roll angle

This graphic presents both the estimated roll angle and the roll setpoint, which is the desired roll angle to fly automatically. As we can see, both are highly correlated, demonstrating the expected behavior.

5.4.3. 2D Trajectory

In this subsection one can observe the 2D trajectory of the flight displayed on top of a grid scale.

6. Conclusions

In closing, we will now detail the achievements of the work performed and provide useful notes for future developments in this topic.

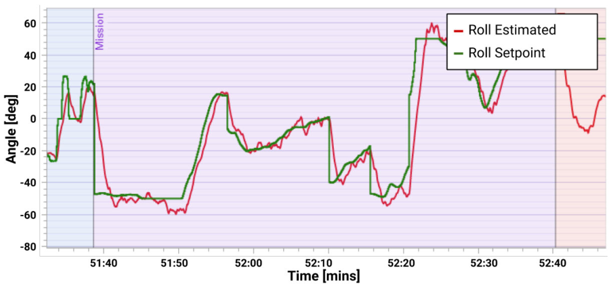

Figure 10: Roll angle.

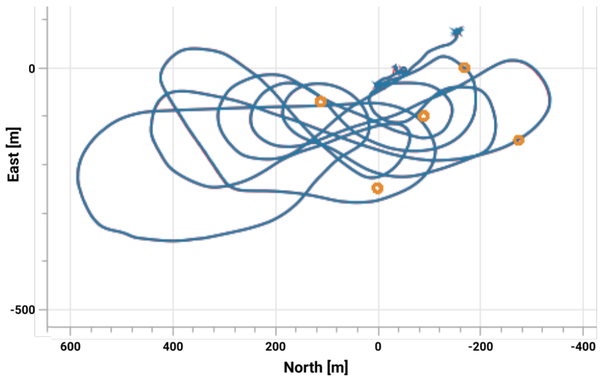

Figure 11: 2D trajectory — Grid scale.

6.1. Achievements

The first DIMA Test Bench was built on COTS components that did not possess the maturity compatible with aeronautic requirements. While this solution was cost effective, this study aimed to improve on the maturity of the avionics platform and, consequently, to increase the representativeness of the demonstrator and its applicability in aeronautic context. For this purpose, the study targeted the following goals: to integrate a tailored DIMA platform in a UAV, providing automated flight functionality and payload driven reconfiguration, and to study and prototype the usage of SDR in DIMA architectures.

To fulfill the first goal, a tailored version of the DIMA platform was integrated in an aircraft model. The DIMA-FLY configuration acts as a validation of DIMA platform's maturity, since it imposes requirements closer to the typical ones for aeronautical systems in real flight scenarios. A Pixhawk flight control system was integrated in the network, and, coupled with a GNC application, provides the aircraft model with automated flight capabilities. With the development of the GNC application, a new Ground Control Station software was also prototyped.

The final goal was to study SDR technologies, investigate possible synergies with DIMA architectures and prototype a data link solution. Using a prototyping board called bladeRFx40, a single-channel data link solution was implemented in the DIMA platform. Using an integrity-checking functionality at each end allows for the verification of data payload. The link also includes a scrambling technique, which encrypts and protects the data being transmitted. These boards connect to the DIMA network using static RDCs and allow any element on the network to send and receive data across the link.

In conclusion, this study was an important and successful step in the development of DIMA as a platform, acting as a validation for its maturity while imposing a real life scenario on this architecture.

6.2. Recommendations for future work

While this study improved certain aspects of the DIMA platform, there are still relevant activities that could be developed in future projects to improve the platform. Some are described below.

- DIMA-FLY's automated flight capabilities are currently dependent on the low-level functionalities of the Pixhawk. It would be interesting to transfer more capabilities to the DIMA processing modules.

- At the same time, the Attitude-target flight mode didn't prove to be mature enough, suspending initial plans for a test flight using this mode. While the simulated implementation proved to be successful, more effort would have to be dedicated to obtain a flight-ready version of the algorithm.

- SDR proved to be a complex field of work, and the prototype displayed some limitations. The data link displayed a small range of around 10 to 20 meters. Due to this limitation, DIMA-FLY used a traditional telemetry data link for the flight scenario. Due to the amount of effort and resources needed to solve this issue, it was left as potential future work.

- In addition, the use of a proper avionics network, perhaps based on Avionics Full-DupleX Switched Ethernet (AFDX), could also be further considered. The reconfiguration of time-predictable networks was considered out of scope of this study. This simplification in terms of network choice has resulted in several issues being avoided and in a simplification of the configuration, as no network configuration was required. Nonetheless, the impacts of the network on the reconfiguration need to be further explored as the distributed system and, therefore, the reconfiguration, heavily depend on the network capabilities.

Acknowledgements

The author would like to thank Cláudio Silva and the rest of GMV's team for providing the opportunity, tools and knowledge needed to perform this study. Another thank you goes to Prof. Agostinho Fonseca, which provided valuable guidance throughout the process.

References

[1] XKY User Guide. GMV, February 2018.

[2] M. T. Barros, J. A. S. Neves, J. R. P. Negrão, S. D. Penna, and M. A. A. Ortiz. Distributed Integrated Modular Avionics. In *STO-MP-IST-166*, number 3. STO, October 2018.

[3] A. Mairaj. Preferred choice for resource efficiency: Integrated modular avionics versus federated avionics. In *2015 IEEE Aerospace Conference*, pages 1–6, March 2015.

[4] S. Park, J. Deyst, and J. How. A new nonlinear guidance logic for trajectory tracking. 08 2004. ISBN 978-1-62410-073-4. doi: 10.2514/6.2004-4900.

[5] K. R. Bruce. Integrated autopilot/autothrottle for the nasa tsrv b-737 aircraft: Design and verification by nonlinear simulation. 03 1989.

[6] G. Wang and Q. Gu. Research on distributed integrated modular avionics system architecture design and implementation. In *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, pages 7D6–1–7D6–10, October 2013.

[7] C. B. Watkins and R. Walter. Transitioning from federated avionics architectures to integrated modular avionics. In *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, pages 2.A.1–1–2.A.1–10, October 2007.

[8] B. Witwer. Systems integration of the 777 airplane information management system (AIMS). *IEEE Aerospace and Electronic Systems Magazine*, 11(4):17–21, April 1996.