VOL.39 Nº1 2022
https://doi.org/10.22491/IMECTA.10838.en

RPAS Applied to the monitoring of areas with different priorities with employment of a ground control station for dynamic operation by task-based interface

Luis Claudio Batista da Silva1*[a,b], Ricardo Maroquio Bernardo*[a,c], Erick Menezes Moreira*[a], Paulo F. F. Rosa*[a]
[a]Military Engineering Institute (IME)
Praça General Tibúrcio, 80, 22290-270, Praia Vermelha, Rio de Janeiro, RJ, Brazil
[b]Celso Suckow da Fonseca Federal Center for Technological Education (CEFET/RJ)
Rua do Imperador, 971, 25620-003, Centro, Petrópolis, RJ, Brazil
[c]Federal Institute of Education, Science, and Technology of Espírito Santo (IFES)
Rodovia BR 482, s/n, 29311-970, Cachoeiro de Itapemirim, ES, Brazil
***luis.silva@cefet-rj.br, *ricardo.maroquio@ifes.edu.br, *emenezes@ime.eb.br, *rpaulo@ime.eb.br**

ABSTRACT: This work presents a system to control multiple small remotely piloted aircrafts. The purpose of the system is to provide autonomous control of aircraft while providing a high-level, task-based user interface that can make possible a dynamic mission control by a single operator. To maintain the situational awareness, aircrafts must form a network, through which the sending of the sensed data by distant aircraft and without reach of the control station can be retransmitted by closer aircraft. For the coverage task, a gradient priority monitoring approach was proposed, by partitioning the area in concentric layers, whose refresh rate varies gradually across the layers. For the validation of the proposed system, simulations and real flights were carried out with a fleet of quadcopters connected to the developed ground control station system. In actual flight experiments, it was possible for a single operator to control a fleet of three quadcopters. By analyzing the flight log data of each aircraft, it was possible to validate the fulfillment of the objectives proposed for the developed system.

KEYWORDS: Remotely piloted aircrafts system. Command and control. Flying ad hoc networks.

RESUMO: Este trabalho apresenta um sistema para controle de múltiplas aeronaves remotamente pilotadas de pequeno porte, aplicado a tarefas de monitoramento por cobertura de área com diferentes prioridades de interesse. O sistema tem por propósito prover um controle autônomo das aeronaves, enquanto fornece uma interface de usuário para supervisão centrada na tarefa, através da qual um único operador possa realizar o controle da missão. Para a manutenção da consciência situacional, as aeronaves devem formar uma rede, através da qual o envio dos dados sensoriados por aeronaves distantes e sem alcance da estação de controle possam ser retransmitidos por aeronaves mais próximas. Para a tarefa de cobertura, foi proposta uma abordagem de monitoramento por prioridades gradientes, através do particionamento da região em camadas concêntricas, cuja frequência de atualização varia gradualmente através das camadas. Para validação do sistema, foram realizados voos com uma frota de três quadricópteros conectados à estação de controle desenvolvida. Pela análise dos logs dos voos, foi possível validar os objetivos propostos para o sistema.

PALAVRAS-CHAVE: Sistema de aeronaves remotamente pilotadas. Comando e controle. Rede ad hoc de veículos aéreos.

1. Introduction

Remotely piloted aircraft systems (RPAS) have been applied in area coverage monitoring for civil and military purposes. Air vehicle coverage is applied to various tasks, such as mapping, patrolling and surveillance, target acquisition, and search and rescue. For the interests of this work, the tasks of persistent monitoring applied to intelligence, surveillance, and reconnaissance missions, commonly performed in defense operations, stand out. This task requires frequently revisiting the points of interest and the monitored region to have points with different priorities, which may change with the evolution of the mission. Updating the sensed information aims to promptly provide the mission control with knowledge of the environment to maintain situational awareness, a fundamental requirement in command and control activities.

The use of RPAS has several advantages since the task environment can be unsafe for the presence of humans, and the tasks in execution can be tedious, leading to fatigue and consequently to the reduction of the mission operators' concentration. While using remotely piloted aircraft (RPA) eliminates the risk of pilot presence in the task environment, solutions that adopt vehicles with critical payloads may require a high level of oversight. On the other hand, small RPAs have been explored for monitoring missions with resources applied only to sensing the environment. Among the advantages of these aircraft, the low cost and lower operating risk stand out. However, resource constraints due to low payload capacity, such as power and communication range, may limit efficacy [1]. The use of multiple aircraft in cooperative work applied to this type of task has been widely investigated to reduce such limitations and increase this type of solution's robustness, efficiency, and reliability [2]. However, such a solution increases the complexity, and several challenges present themselves, such as control and integration.

The objective of this work was to develop a system to control multiple small-sized RPAs, applicable to the monitoring of areas with different priorities of interest and dynamic allocation of tasks assigned by a single operator. The architecture of the proposed solution is divided into three parts: (a) a ground control station for dynamic mission planning and control through a task-based user interface; (b) a fleet of quadcopters capable of autonomous navigation by inertial sensors and satellites; (c) control software embedded in the aircraft, responsible for the communication management, allocation of trajectories for the aircraft and control of the sensors applied in the mission.

2. Literature Review

This section presents the theoretical foundations for remotely piloted aircraft systems and compares topic-related studies.

2.1 Remotely Piloted Aircraft System

According to the National Civil Aviation Agency (ANAC) [3], a remotely piloted aircraft system (RPAS) is a system comprising not only the remotely piloted aircraft (RPA) but also the necessary infrastructure for its operation, such as a remote pilot station, command, and control link, launch and recovery equipment, among others.

- **Figure 1** presents a high-level view of the architecture of an RPAS. The main components of this architecture and their functions are presented below:
- Ground control station (GCS): prepare and send flight plans to the RPAs, with a graphical interface for mission control and visualization of sensed data;
- Manual radio control: redundancy requirement for operator intervention in case of system failures; the onboard receiver decodes the control commands;
- Flight controller: run an autopilot software to control the aircraft's flight attitude; receive commands from the radio control or messages via the data radio (in a defined protocol); perform the reading of the inertial sensors (INS) and satellite receiver (GPS) to generate the necessary signals for the actuators, aiming to fulfill the required flight attitude;
- Onboard computer: in addition to the flight controller, a microcomputer can be embedded in the RPA, which, integrated with the autopilot, allows the processing of information sensed and the reading of onboard sensors and may even assist in navigation;
- Onboard software: software that runs on the onboard computer; it works integrated with the autopilot, sending commands and receiving data readings from the sensors;

Power module: supply the energy necessary for the operation of the aircraft components.

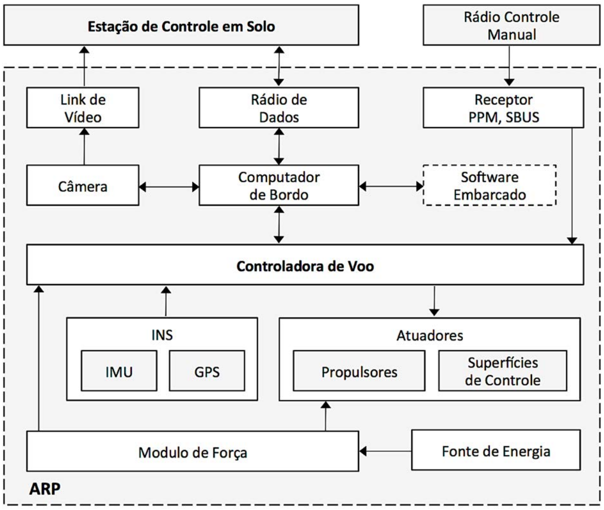


Fig. 1 – High-level schematic of RPAS components.

2.2 Classification of RPAs

- According to DECEA [4], RPAs can be classified regarding the level of flight automation as:
- Remotely piloted: the pilot's action is direct during all phases of the flight;
- Automatic: they work with an automatic onboard pilot so that they can follow the planned flight path without human intervention. However, it allows the pilot to interfere at any time;

Autonomous: performs a previously embedded flight plan from start to finish, from takeoff to landing, without allowing a pilot to interfere or change the parameters initially established for the flight.

Regarding the classification by size, according to ANAC regulations [5], the RPAs can be classified per the maximum takeoff weight (**Table 1**).

Tab. 1 – Classification of RPAs according to ANAC.

Class	Characteristics
Class 1	Maximum takeoff weight greater than 150 kg
Class 2	Maximum takeoff weight greater than 25 kg and up to 150 kg
Class 3	Maximum takeoff weight up to 25 kg

There is also the classification of aircraft in terms of lift. It considers the structural characteristics and properties related to the flight mode. The main classifications by type of lift are fixed-wing (airplane), rotary wing (helicopter and multirotor), lighter than air (airship), among others.

2.3 Quadricopter

Quadcopters are rotary wing-type RPAs, powered by four rotors powered by electric motors and controlled by an electronic control system for flight stabilization. They have vertical takeoff and landing (VTOL) capability, hovering capability, and good maneuverability. It is a nonlinear system, tightly coupled with 6-DOF (degrees of freedom), with three linear and three angular motions. The forces and moments are generated by the propellers attached to the rotors. Therefore, the vehicle is controlled by the angular speeds of the engines, which produce a thrust and a torque that, combined, generate a main thrust, and the roll (Φ), pitch (θ), and yaw (ψ) [6] torques. **Figure 2** presents the angular movements (Φ, θ, ψ) concerning the axes (x,y,z) of the reference system fixed to the vehicle. The weight force is given by *mg*.

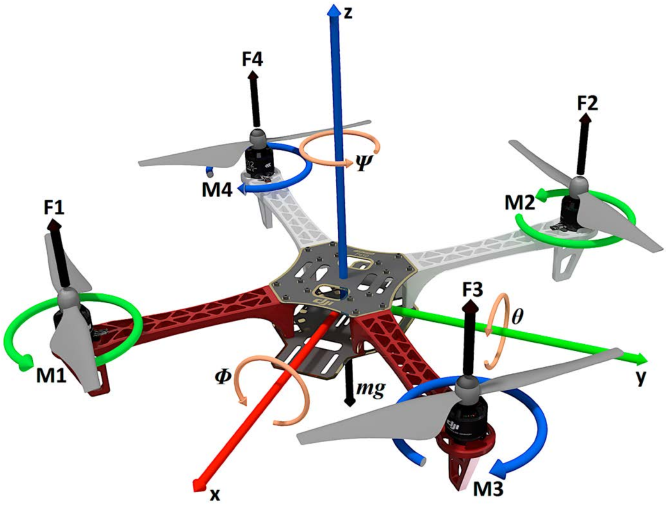


Fig. 2 – Forces (Fi) and moments (Mi) of the quadcopter.

2.4 Related Studies

The proposed system's objective is to monitor regions with points with different priorities of interest. Considering the decomposition of a region under monitoring into cells, [7] presents the concept of cell "*age*" as the time interval since the last update during its detection by an aircraft. Thus, a cell with a higher priority must have its maximum age lower than the one with a lower priority. In [8][9][10], the authors present solutions in which PRAs work autonomously to monitor regions with different priorities of interest. However, unlike the system proposed in this work, allowing the dynamic assignment of priorities during the mission, priorities are defined *a priori* in the works investigated. They cannot be redefined during the mission.

Another requirement of the present work is optimizing human resources employed in the mission. In [11], the authors discuss the overload on a single pilot when he needs to control several aircraft tasks. They present a comparison between vehicle-based control, where the operator assigns tasks individually to each RPA, and task-based control, in which the operator assigns a list of tasks to the fleet of RPAs, which autonomously distributes these tasks between aircraft. The authors indicate significant advantages in task-based control, both in performance and in the robustness of the solution. However, they point out that task-based control can lead to a loss of control in unexpected situations. In [12][13], the authors present task-based control solutions in which a single pilot commands the mission through a high-level task assignment interface. However, communication between the ground station and the vehicles is done individually. In the present work, communication must occur through a network formed by the aircraft. Therefore, it increases the robustness and coverage of the system.

Communication restrictions can also reduce the effectiveness of small RPAs in monitoring tasks. Research in [14][15] presents the collaborative use of multiple aircraft in surveillance tasks, in which communication constraints are dealt with by forming a network between RPAs. Recent research [16][17][18] addresses the problem of communication between multiple RPAs as a new family of network, called *ad hoc* aircraft network (FANET), seen as a unique form of *ad hoc* networks of mobile devices (MANET) and *ad hoc* vehicle networks (VANET). The authors indicate that networks formed by aircraft need new approaches due to the challenges imposed both by the distance of coverage and by the mobility of these vehicles, which require constant changes in the network topology.

The contribution of the present work is the elaboration of the architecture of a robust system for the autonomous control of a fleet of RPAs, connected to a ground control station, provided with a high-level supervision interface to support the dynamic control of the task-based mission by single operator control.

3. Problem Statement

The proposed system is applicable to area-coverage monitoring, in which sensed information must be delivered to the mission control station promptly to maintain situational awareness so that it is a portable and ready-to-use command and control resource. Thus, the RPAS must comprise small RPAs and a compact GCS.

Thus, we propose to observe a point of interest, maintaining a security perimeter as a task performed by the RPAS. The priority decays proportionally to the radial distance from the center point of most significant interest to the region's edges under monitoring.

We propose to use multiple RPAs for coverage to partition the region of interest between several aircraft in cooperative work. Aircraft must form a data network so that data sensed by an aircraft whose communication capabilities do not reach the control station can be relayed by other, closer aircraft.

Also, for RPAs to be dynamically controlled by a single operator, the GCS must provide a high-level user interface for task-based control, with an interactive mapping system for mission planning, control, and tracking. The tasks requested by the operator must be planned by the system and distributed among the RPAs to fulfill the flight plans autonomously. The GCS must also provide controls so that the operator can intervene in the functioning of any RPA to fulfill basic navigation functions.

3.1 Application Domain

The main elements of the application domain for the proposed task for the RPAS are described below:

- The W environment is outdoor and has a set of coordinates$\{C_1, C_2,..., C_n\}$. Each $C_i \varepsilon R^3$ is a point on the Earth's surface [*latitude*, *longitude*, *altitude*];
- The area of interest A under monitoring is a subset of W, which forms a closed polygon;
- P is a partition of A, and each element $c_j \varepsilon P$ is called a "*cell*," whose size l is the sensor range;
- The center cell c_0 of A is the highest priority point; the other cells of A form a perimeter with K concentric layers around c_0; the priority of each layer decays proportionally to the distance from c_0;
- The priority of interest determines the update frequency f_k of cell c_j in layer k, for $k \leq K \varepsilon N$; cells of the same layer have the same priority;

- Fleet U is formed by n autonomous aircraft $\{u_1, u_2, \ldots, u_n\}$; each u_i at time t monitors layer k.

For this work, the following restrictions are considered: the region under monitoring is free of obstacles, RPAs are similar concerning flight capability, and the sensors embedded for the execution of the mission are homogeneous in terms of the sensing range.

4. Proposed Solution for the RPAS

For the area coverage task, a hexagonal grid was adopted due to the property of the same navigation distance for any neighboring cell. We propose a "*priority gradient*" approach to monitoring areas with different interest priorities. The center cell (c_0) of the gradient has the highest priority. The other cells form concentric layers around c_0, whose priority decreases proportionally towards the edge of the gradient (see **Figure 3**). Cells in the same layer have the same priority and must be revisited with the same frequency.

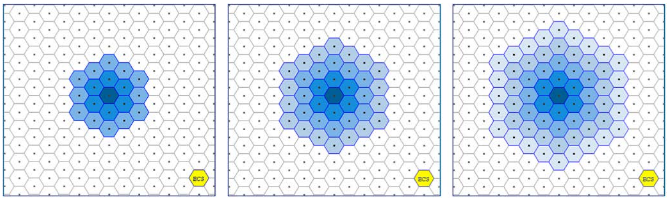


Fig. 3 – Priority Gradient with 3, 4, and 5 layers.

Considering a fleet of aircraft $U = \{u_1, u_2, \ldots, u_n\}$ with homogeneous sensors, the total coverage area A_{max} of the priority gradient over the hexagonal grid, considering the size of the side l of the hexagon based on the sensor range and the gradient with K layers, is given by:

$$A_{\max} = \frac{3\sqrt{3}}{2} l^2 \left(\left(\sum_{k=1}^{n} 6(k-1) \right) + 1 \right) \qquad (1)$$

The frequency f_k of revisits for each cell in layer k, considering the coverage of each layer by only one aircraft, in which va is the aircraft speed, is given by:

$$f_k = \frac{1}{\frac{\sqrt{3l}}{v} 6(k-1)}, \forall k \geq 2 \in N \qquad (2)$$

4.1 Proposed Reallocation Algorithm

When starting the mission, the aircraft occupy the gradient layers in order of priority. During the mission, the operator can request to relocate the gradient position. At this time, the RPA fleet should receive the new plans to occupy the new position of interest.

A trivial solution would be to move the RPAs to the same layer they were in before the reallocation. However, this solution does not prioritize the occupation of the center cell (higher priority) since there may be RPA without external layers closer to the center of the new gradient, which would occupy this position in a shorter time. The proposed approach is the coordination of agents by strategic positioning [19]. Here, the RPA closest to the center of the new gradient position will occupy this position. This strategy applies to all other aircraft so that the other layers will also be occupied by the closest RPA in descending order of priority, as presented in [20].

4.2 Basic Architecture of the Proposed RPAS

This work proposes a modular architecture for an RPAS, as shown in **Figure 4**. The aircraft fleet is controlled by an autopilot capable of autonomous navigation by an inertial system and satellites. The autopilot receives flight plans from an onboard control system, responsible for allocating tasks, controlling the payload, and managing communication resources. At the control station, resources for mission planning and a user interface for mission control are available.

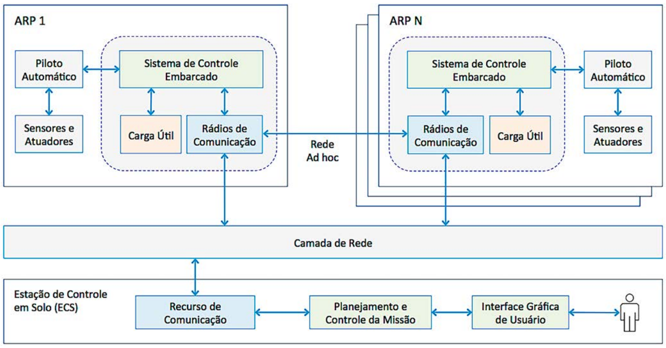


Fig. 4 – Overview of the proposed RPAS architecture.

For the onboard control system, a layered architecture was designed with different levels of abstraction, in which each layer acts as a client for the lower layer and as a server for the upper layer (**Figure 5**). Each GCS demand must be handled by the layers, decreasing the level of abstraction as it approaches the autopilot system. At the highest level of the architecture, the GCS sends commands to the onboard control and receives telemetry over a task-based communication protocol. The transfer of control to the autopilot adopted the strategy of implementing a middleware whose objective is to reduce the level of abstraction of the application to control the vehicle. The autopilot adopted is the ArduCopter, which uses the MAVLink (Micro Air Vehicle Communication Protocol) communication protocol. Thus, the Dronekit API (open-source) was used, which offers an intermediate level of abstraction for MAVLink communication, receiving commands by object messages and performing the conversion to the protocol format.

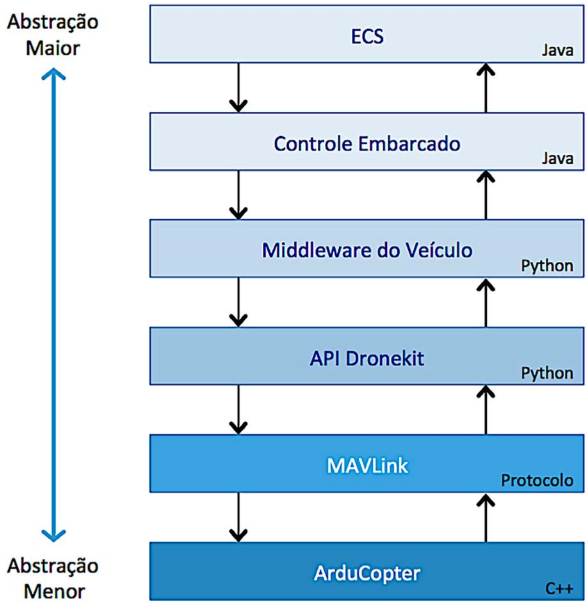
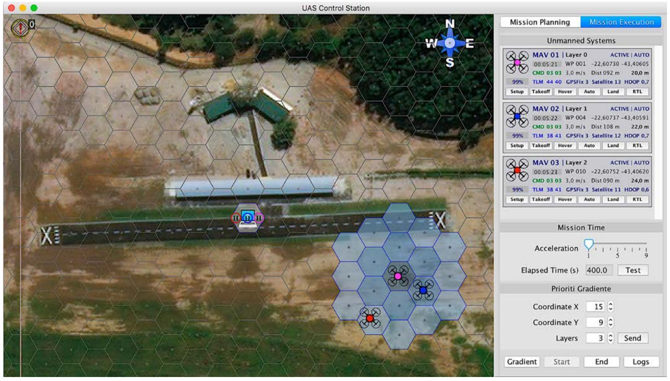


Fig. 5 – Basic diagram of integration of the RPAS.

4.3 Ground Control Station

The RPAS has a GCS with a supervisory-level graphical user interface based on the task, i.e., an interface in which the operator can accomplish the mission objectives without engaging with the aircraft. The operator interacts with the map of the region where the task takes place to command the mission. The system distributes the tasks among the aircraft, which perform the flight in autonomous mode. **Figure 6** presents the developed GCS graphical interface.

The control panel has two tabs: one for planning and another for the mission's dynamic control (execution). The control tab displays real-time telemetry information for each RPA connected to the GCS.



MAV 01 | Layer 0 ACTIVE | AUTO
00:05:21 WP 001 -22,60730 -43,40605
CMD 03 03 3,0 m/s Dist 092 m 20,0 m
99% TLM 44 40 GPSFix 3 Satellite 13 HDOP 0,7

MAV 02 | Layer 1 ACTIVE | AUTO
00:05:22 WP 004 -22,60737 -43,40591
CMD 03 03 3,0 m/s Dist 108 m 22,0 m
99% TLM 38 41 GPSFix 3 Satellite 12 HDOP 0,7

MAV 03 | Layer 2 ACTIVE | AUTO
00:05:23 WP 010 -22,60752 -43,40620
CMD 03 03 3,0 m/s Dist 090 m 24,0 m
99% TLM 38 41 GPSFix 3 Satellite 11 HDOP 0,6

Acceleration 1 5 9
Elapsed Time (s) 400.0
Coordinate X 15
Coordinate Y 9
Layers 3

Fig. 6 – Graphical user interface of the RPAS GCS.

Figure 7 shows the telemetry panel available for each aircraft, as described below:

- (A) RPA Id and RPA allocation gradient layer;
- (B) Total flight time of the aircraft applied in the mission;
- (C) Counter of commands sent by the GCS and received and executed by the onboard control system;
- (D) Battery level and telemetry counter sent by the onboard control system and the autopilot;
- (E) Aircraft communication link configuration;
- (F) Autopilot status and RPA flight mode;
- (G) Waypoint index, latitude, and longitude of the position;
- (H) Aircraft speed, GCS distance, and altitude;
- (I) GPS conditions: number of fixed satellites (minimum of 3 required); visible satellites; dilution index of the horizontal geometry of the satellites (ideal<1);
- (J) Navigation commands that can be sent to the RPA to perform outside the mission; can be used by the operator to take immediate control of the RPA.

Telemetry and command counters are used so that the system can validate message deliveries without the need for a communication protocol confirmation mechanism.

Fig. 7 – Real-time telemetry panel for each RPA.

4.4 Construction of the Flight Platform

For the RPAS aircraft fleet, the use of quadcopters was adopted. This type of aircraft meets the system requirement for a portable solution. The quadcopters have a platform with simple control mechanisms, vertical takeoff and landing (VTOL), hovering capability, and a wide range of components available on the market. **Figure 8** shows a basic scheme for integrating the quadcopter components. **Table 2** presents the specifications of the quadcopters built for the RPAS.

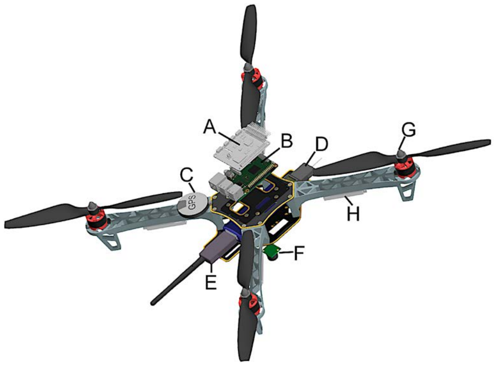


Fig. 8 – Integration scheme of the quadcopter components: (A) Navio2 flight controller; (B) RaspberryPi 3B; (C) GPS antenna; (D) 2.4 GHz hand control receiver; (E) XBee Pro 900 MHz data radio; (F) MaxBotix I2CXL-EZ0 sonar; (G) 920 kV 2312 brushless motor and 9.45" propellers; (H) 20 A electronic speed controller.

The components can be grouped by function, such as propulsion components (engines; electronic speed controllers - ESC; propellers), processing components (flight controller; onboard microcomputer), communication components (radio control receiver; data radio; WiFi connection), in addition to the chassis (frame), which provides a rigid structure to the vehicle and support for fixing the other components. The chassis used is of type X: the front of the aircraft is pointed by a pair of engines that rotate in the opposite direction, using the X-axis of the North-East-Down coordinate system as a reference.

The flight controller adopted for the project was Navio2 from Emlid [21]. Unlike most controllers, Navio2 does not process the autopilot software directly. However, it works as a component integrated into the GPIO bus of RaspberryPi 3B, with ArduPilot [22] being the main (open-source) autopilot software supported. However, Navio2 only uses part of the processing power of RaspberryPi 3B so that other applications can run in the same autopilot environment.

Tab. 2 – Main specifications of the quadcopter.

Chassis size	450 mm
Total takeoff weight	1350 g
Extra load capacity	Approximately 800 g
Flight time	8-17 min (depends on extra load)
Flight controller	Navio2 (Emlid)
Autopilot	ArduCopter 3.4 (ArduPilot)
Battery	3-cell LiPo; 5200 mAh; 10C

5. Experiments and Results

The validation of the proposed RPAS used the GCS developed for planning and control of the multiple autonomous flights and the control software embedded in the RPAs to allocate the commands and plans received in real-time from the GCS. Thus, three quadcopters with similar hardware were used (**Figure 9**), with the autopilot parameters configured according to **Table 3**.

Fig. 9 – Equipment used in the flight experiment.

Tab. 3 – Configuration of the parameters of the quadcopters.

Parameter	RPA (1)	RPA (2)	RPA (3)
Horizontal Speed	3.0 m/s	3.0 m/s	5.0 m/s

Parameter	RPA (1)	RPA (2)	RPA (3)
Vertical Speed - Climb	2.5 m/s	2.5 m/s	2.5 m/s
Vertical Speed - Descent	1.5 m/s	1.5 m/s	1.5 m/s

Flights were performed using an *ad hoc* network using XBee radios. Due to the developed GCS, no flight plans were preloaded on the aircraft, and only the control software was embedded. The distance between the centroids of the priority gradient cells was parameterized to 12.5 m.

Figure 10 shows the flight path performed by each aircraft to cover the priority gradient. The aircraft can change layers in the proposed relocation algorithm to optimize the gradient occupation time. The map shows that RPA 1 starts by covering layer 2 (**Figure 10a**). However, when the priority gradient was reallocated to another area, it was allocated to layer 1, i.e., the center of the gradient (**Figure 10b**).

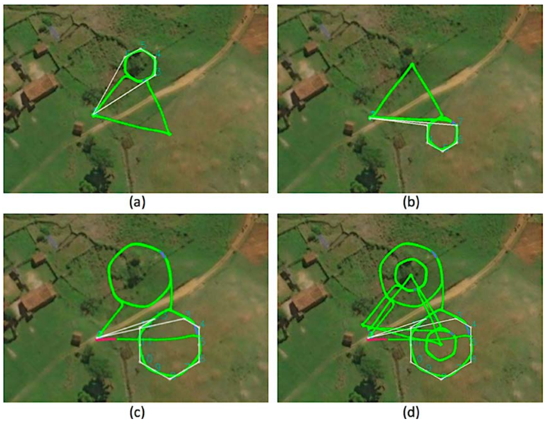


Fig. 10 – Map of aircraft trajectories in Multiple Flight: (a) trajectory of RPA 1; (b) trajectory of RPA 2; (c) trajectory of RPA 3; (d) overlapping of RPA trajectories.

Figure 11 presents graphs of the position of the trajectories executed by each RPA while performing the task. The different time points of the flight are highlighted in the graph, from takeoff to landing. We can also observe the time point when the commands are sent by the GCS (GCS cmd) to the RPAs. For example, **Figure 10a** shows the trajectory of RPA 1, which starts at layer 2 of the first position of the priority gradient. After relocating the gradient to the second position requested by the operator, it switches to layer 1 to meet the priority of the center cell because it is closer to this point at the time of gradient reallocation.

Figure 12 presents frames extracted from the video recorded during the autonomous flight experiment controlled by a single operator. The frames are in order of events commanded by the operator, from takeoff to landing, with the relocation of the priority gradient position. Next to each frame is the GCS interface's screenshot during the flight. It highlights the display of the position and telemetry of each aircraft at the actual time of the task.

Table 4 presents the data from analyzing the flight logs generated by the onboard autopilot in each quadcopter. For RPAs (1) and (2), the position error of the trajectory performed, compared with the trajectory performed by the aircraft, was sub-metric, considered a satisfactory result for the use of navigation by the satellite system. We can observe that the RPA (3), configured for a horizontal speed two m/s higher than the others and which covered layer 3 of the priority gradient, presented a more significant position error than the other aircraft. The experiments showed that the higher the RPA speed, the greater the position error concerning the vertices of the flight path.

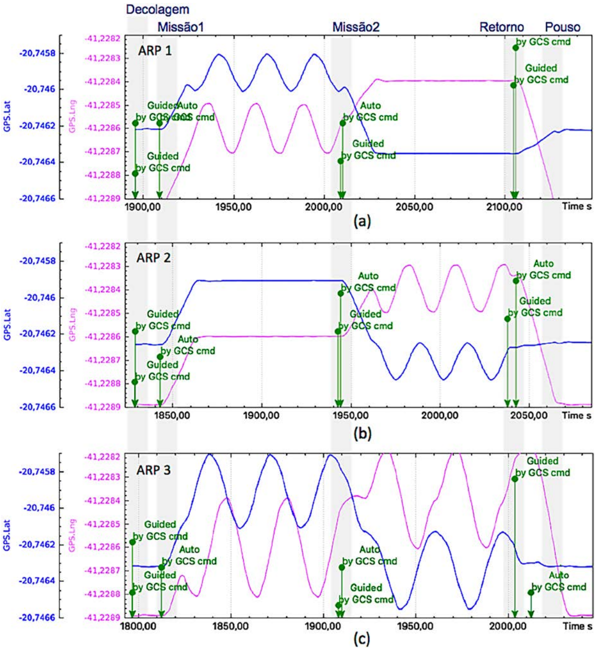
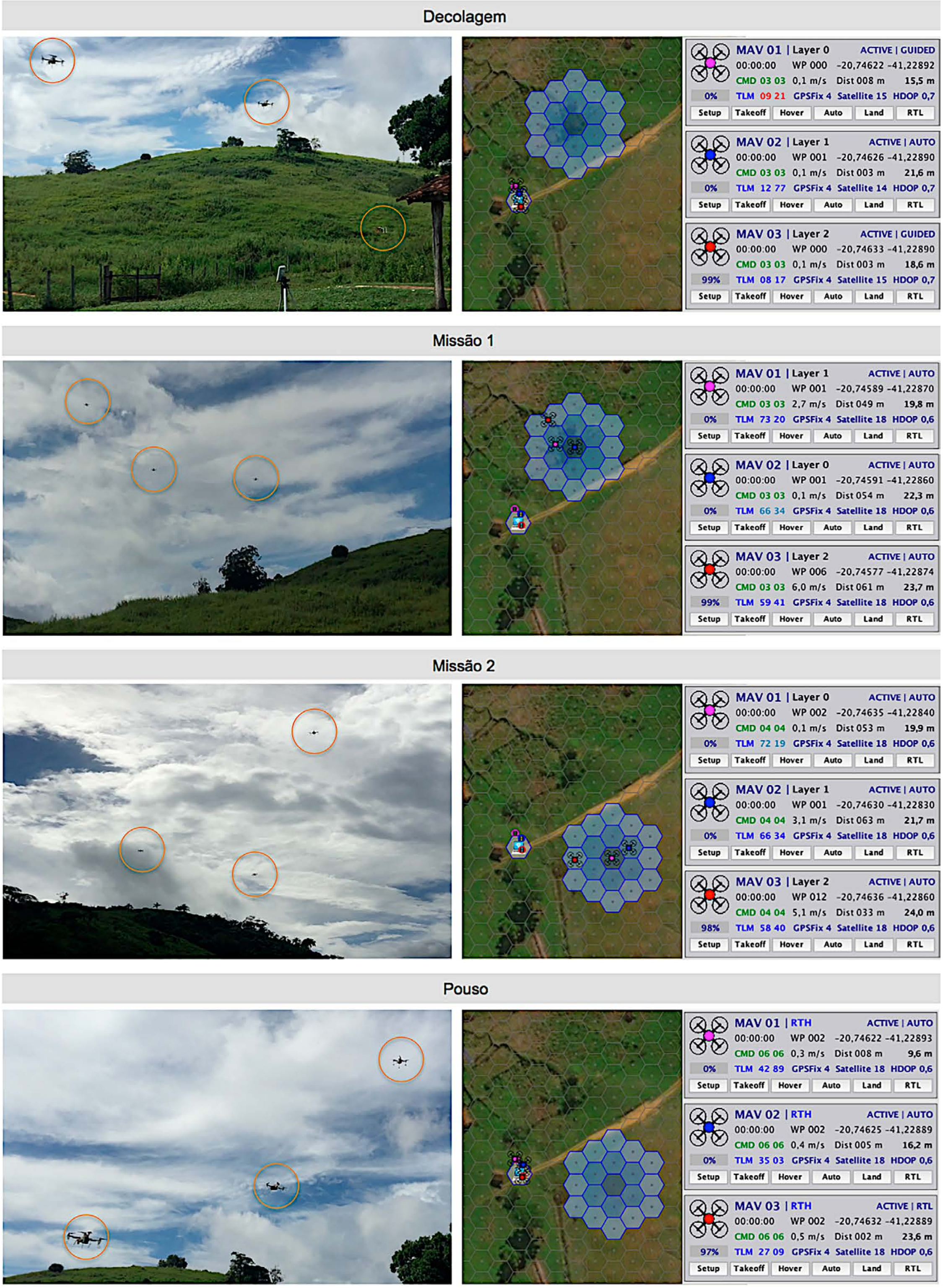


Fig. 11 – Graphs of the trajectories performed by the quadcopters for the multiple real flight experiment: (a) latitudes and longitudes of RPA 1; (b) RPA 2, and (c) RPA 3.



```
MAV 01 | Layer 0        ACTIVE | GUIDED
00:00:00     WP 000  -20,74622 -41,22892
CMD 03 03  0,1 m/s  Dist 008 m    15,5 m
0%  TLM 09 21  GPSFix 4  Satellite 15  HDOP 0,7

MAV 02 | Layer 1          ACTIVE | AUTO
00:00:00     WP 001  -20,74626 -41,22890
CMD 03 03  0,1 m/s  Dist 003 m    21,6 m
0%  TLM 12 77  GPSFix 4  Satellite 14  HDOP 0,7

MAV 03 | Layer 2        ACTIVE | GUIDED
00:00:00     WP 000  -20,74633 -41,22890
CMD 03 03  0,1 m/s  Dist 003 m    18,6 m
99%  TLM 08 17  GPSFix 4  Satellite 15  HDOP 0,7
```

```
MAV 01 | Layer 1          ACTIVE | AUTO
00:00:00     WP 001  -20,74589 -41,22870
CMD 03 03  2,7 m/s  Dist 049 m    19,8 m
0%  TLM 73 20  GPSFix 4  Satellite 18  HDOP 0,6

MAV 02 | Layer 0          ACTIVE | AUTO
00:00:00     WP 001  -20,74591 -41,22860
CMD 03 03  0,1 m/s  Dist 054 m    22,3 m
0%  TLM 66 34  GPSFix 4  Satellite 18  HDOP 0,6

MAV 03 | Layer 2          ACTIVE | AUTO
00:00:00     WP 006  -20,74577 -41,22874
CMD 03 03  6,0 m/s  Dist 061 m    23,7 m
99%  TLM 59 41  GPSFix 4  Satellite 18  HDOP 0,6
```

```
MAV 01 | Layer 0          ACTIVE | AUTO
00:00:00     WP 002  -20,74635 -41,22840
CMD 04 04  0,1 m/s  Dist 053 m    19,9 m
0%  TLM 72 19  GPSFix 4  Satellite 18  HDOP 0,6

MAV 02 | Layer 1          ACTIVE | AUTO
00:00:00     WP 001  -20,74630 -41,22830
CMD 04 04  3,1 m/s  Dist 063 m    21,7 m
0%  TLM 66 34  GPSFix 4  Satellite 18  HDOP 0,6

MAV 03 | Layer 2          ACTIVE | AUTO
00:00:00     WP 012  -20,74636 -41,22860
CMD 04 04  5,1 m/s  Dist 033 m    24,0 m
98%  TLM 58 40  GPSFix 4  Satellite 18  HDOP 0,6
```

```
MAV 01 | RTH              ACTIVE | AUTO
00:00:00     WP 002  -20,74622 -41,22893
CMD 06 06  0,3 m/s  Dist 008 m     9,6 m
0%  TLM 42 89  GPSFix 4  Satellite 18  HDOP 0,6

MAV 02 | RTH              ACTIVE | AUTO
00:00:00     WP 002  -20,74625 -41,22889
CMD 06 06  0,4 m/s  Dist 005 m    16,2 m
0%  TLM 35 03  GPSFix 4  Satellite 18  HDOP 0,6

MAV 03 | RTH               ACTIVE | RTL
00:00:00     WP 002  -20,74632 -41,22889
CMD 06 06  0,5 m/s  Dist 002 m    23,6 m
97%  TLM 27 09  GPSFix 4  Satellite 18  HDOP 0,6
```

Fig. 12 – Video frames of the actual flight experiment and a screen capture of the ground control station, highlighting the aircraft's position on the map and the real-time telemetry of each RPA for each phase of the flight. Takeoff: Beginning of the mission with automatic takeoff. Mission 1: the occupation of the first area of interest. Mission 2: reallocation of the aircraft fleet (after operator request) to occupy the second area of interest. Landing: after the operator's request to end the mission, the aircraft fleet return to the takeoff point and land automatically.

Tab. 4 – Result of analysis of multiple flight logs.

	Measured Parameter	RPA (1)	RPA (2)	RPA (3)
Layer 1	Average horizontal speed	1.36 m/s	1.41 m/s	---
	Average position error	0.28 m	0.19 m	---
	Mean absolute position deviation	0.21 m	0.13 m	---
Layer 2	Average horizontal speed	2.45 m/s	2.28 m/s	---
	Average position error	0.64 m	0.65 m	---
	Mean absolute position deviation	0.28 m	0.33 m	---
Layer 3	Average horizontal speed	---	---	3.47 m/s
	Average position error	---	---	1.31 m
	Mean absolute position deviation	---	---	0.85 m

Considerations about the flight experiment: this work enabled the validation of the RPAS in the control of multiple RPAs controlled by a GCS, in turn, commanded by a single operator. Only resources developed for the solution were used without the need for third-party software for the planning and execution of the tasks. While control radios were present and configured to take over each aircraft in a system failure, they were not needed during flights. The functionalities of the primary navigation control panel of each RPA, available for the operator to send control commands to any aircraft at any time, were also tested separately. However, no incidents required the use of this resource during the missions.

6. Conclusion

This work proposed a system for coordinating multiple small RPAs, covering areas with different interest priorities, applicable to monitoring tasks. Solutions were investigated, emphasizing autonomous planning and execution of tasks and the availability of GCS with a high-level task-based user interface.

The RPAS project was based on the layered style, aiming at a modular solution, easy to integrate, modify, and adapt to new functionalities. The “priority gradient” approach was proposed to cover areas with different priorities. Here, the region is divided into cells that form concentric layers. The center cell must be covered with higher priority, gradually reducing the priority across the layers. An *ad hoc* network was established using XBee radios to link the RPAs and the GCS.

The developed RPAS was validated using quadcopters built for the research in actual flight experiments. A single operator could control a fleet of three quadcopters in autonomous flight without operator intervention to control any vehicle individually during the mission. The flight logs were captured and analyzed. The RPAs executed the trajectories planned by the GCS. The results of the data analysis provided the validation of the proposed requirements for the system.

Bibliographic References

[1] CHEN, Y.; ZHANG, H.; Xu, M. The coverage problem in UAV network: a survey. **Fifith Internacional Conference on Computing Communication and Networking Technologies (ICCCNT)**, págs. 1–5, 2014.

[2] NIGAM, N.; The Multiple unmanned air vehicle persistent surveillance problem: a review**. Machines**, 2(1):13–72, 2014.

[3] ANAC, Agência Nacional de Aviação Civil. **Sistema de Aeronaves Remotamente Pilotadas**, 2015; http://www2.anac.gov.br/rpas,acessada em maio 2018.

[4] DECEA. Departamento de Controle do Espaço Aéreo. **Voos de RPAS**, 2015; https://www.decea.gov.br, acessada em maio de 2018.

[5] ANAC, Agência Nacional de Aviação Civil.**Classes de drones**; 2017;http://www.anac.gov.br/assuntos/paginas-tematicas/drones/classes-de-drones, acessada emmaio 2018.

[6] CARRILLO, L. R. G.; LÓPEZ, A. E. D.; LOZANO, R.; Pégard, C.; Quad rotorcraft control: vision-based hovering and navigation. **Springer Science & Business Media**, 2012.

[7] NIGAM, N.; Kroo, I.; Persistent surveillance using multiple unmanned air vehicles. **Aerospace Conference 2008 IEEE**, pp. 1–14, 2008.

[8] KHANDANI, H.; MORADI, H.; PANAH, J. Y.; A real-time coverage and tracking algorithm for UAVs based on potential field. Robotics and Mechatronics (ICRoM), 2014 **Second RSI/ISM International Conference**, pp. 700–705., 2014.

[9] HOWDEN, D. J.; Fire tracking with collective intelligence using dynamic priority maps. **Evolutionary Computation (CEC), 2013 IEEE Congress**, pp.2610– 2617. , 2013.

[10] MICHAEL, N.; STUMP, E.; MOHTA, K. Persistent Surveillance with a Team of MAVs**. Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference** pp.2708–2714., 2011.

[11] CUMMINGS, M. L.; BERTUCELLI, L. F.; MACBETH, J.; SURANA, A. Task versus vehicle-based control paradigms in multiple unmanned vehicle supervision by a single operator. **IEEE Transactions on Human--Machine Systems**, 44 (3):353–361, 2014.

[12] ARAUJO, J.; SUJIT, P.; SOUSA, J. Multiple UAV area decomposition and cove- rage. **Computational Intelligence for Security and Defense Applications (CISDA), 2013 IEEE Symposium**, pp.30–37, 2013.

[13] LICHTENSTERN, M.; ANGERMANN, M.; FRASSL, M.; IMU-and GNSS-assisted single-user control of a MAV-swarm for multiple perspective observation of outdoor activities. **ION ITM**, 2011.

[14] ACEVEDO, J. J.; ARRUE, B. C.; MAZA, I.; OLLERO, A. Distributed approach for coverage and patrolling missions with a team of heterogeneous aerial robots under communication constraints. **Int Jounal Adv Robotic Sy**, 10(28), 2013.

[15] ACEVEDO, J. J.; ARRUE, B. C.; DIAZ-BANEZ, J. M.;VENTURA, I.;MAZA, I.; OLLERO, A.; One-to-one coordination algorithm for decentralized area partition in surveillance missions with a team of aerial robots. **Journal of Intelligent & Robotic Systems**, 74(1-2):269–285, 2014.

[16] BEKMEZCI, I.; SAHINGOZ, O. K.; TEMEL, Ş.; Flying ad-hoc networks (FANETs): a survey. **Ad Hoc Networks**, 11(3):1254–1270, 2013.

[17] SAHINGOZ, O. K.; Networking models in flying Ad-hoc networks (FANETs): concepts and challenges. **Journal of Intelligent & Robotic Systems**, 74(1-2):513– 527, 2014.

[18] BEKMEZCI, I.; SEN, I.; ERKALKAN, E.; Flying ad hoc networks (FANET) test bed implementation. **Recent Advances in Space Technologies (RAST), 2015 7th International Conference**, pp.665–668, 2015.

[19] REIS, L. P.;**Coordenação em sistemas multi-agente: aplicações na Gestão universitária e futebol robótico** Tese (Doutorado em Engenharia), Universidade do Porto, Portugal, 2003.

[20] SILVA, L. C. B.; BERNARDO, R. M.; OLIVEIRA, H. A.; ROSA, P. F. F. Multi-UAV agent-based coordination for persistent surveillance with dynamic priorities. **Military Technologies (ICMT), 2017 International Conference**, pp. 765-771, 2017.

[21] EMLID; **Navio2 autopilot hat for Raspberry Pi powered by ArduPilot and ROS**; 2017. https://emlid.com/navio, acessada em janeiro de 2017

[22] ARDUPILOT DEV TEAM; **ArduPilot autopilot suite**, 2017. http://ardupilot.org, acessada em janeiro de 2017.