The BIG Agent: The Utilization of Embedded Multi-Agent Systems in Unmanned Aerial Vehicles for Autonomous Operations

Kauã Santos, Daniel Costa, Nilson Lazarin, Bruno Freitas, Carlos Pantoja

Bacharelado em Sistemas de Informação
Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (Cefet/RJ)
Nova Friburgo, RJ – Brazil

chon@grupo.cefet-rj.br

Abstract. *Inspired by the high crime rates in Brazil and the use of drones in strategic sectors, this work presents an integration of multi-agent systems (MAS) embedded within unmanned aerial vehicles (UAVs) for autonomous and distributed route operations. The system uses a UAV with two embedded MASs, one of them controlling the drone itself. The other one, dubbed Secretary, is responsible for setting routes for the drone. The approach is validated with a simulation where the drone must visit a predetermined set of points, showing it successfully manages to do so.*

1. Introduction

Public security is a critical issue for society that daily affects millions of persons [CNN Brasil 2025]. In major cities like Rio de Janeiro and São Paulo, reported crimes include 58,574 thefts and 30,934 vehicle thefts [SSP-RJ 2025]. Meanwhile, the implementation of security systems faces dual challenges, including infrastructure deficiencies and economic constraints [World Bank 2016]. Technological innovations—particularly smart cameras and sensor-equipped UAVs—are emerging as promising solutions to these surveillance gaps [AL-Dosari and Fetais 2023].

The use of Unmanned Aerial Vehicles (UAVs), which are remotely controlled and provide greater operational safety and flexibility [Osmani and Schulz 2024], is a key example of this technological shift. However, the effectiveness of these and other smart technologies is often limited by the inherent challenges of traditional centralized systems. Regardless of the application, from security to logistics, these centralized architectures face problems such as: i) data overload, due to exponential growth in data volume; ii) low scalability, caused by the high frequency of updates required for the server infrastructure; and iii) high maintenance costs, resulting from the investment needed to maintain and ensure continuous long-term operations [Petritoli et al. 2018].

Despite these challenges, the use of embedded technologies such as Multi-Agent Systems (MAS) enables the management of simultaneous and intelligent operations for multiple UAVs through its distributed approach. These drones can perceive their surroundings and perform actions collaboratively and autonomously [Younes et al. 2016].

An Embedded Multi-Agent System (MAS) consists of autonomous agents that operate cooperatively within a specific device to achieve coordinated objectives

[Pantoja et al. 2023]. The ARGO agent architecture [Pantoja et al. 2016] enables seamless integration with sensor networks and geolocation systems (e.g., GPS), while the Communicator architecture [Jesus et al. 2019] facilitates distributed message exchange between MAS instances. Together, these architectures support decentralized networks of smart devices capable of: (1) cross-platform data sharing, (2) real-time public safety event detection, and (3) automated alert triggering.

Therefore, this study proposes a method for creating a Multi-Agent Security System by the conjunction of an UAV and two Embedded MAS's. This system uses UAVs to intervene in urban environments by establishing a real-time action and data feedback loop with a Security Secretary. This physical-technological integration resembles the fictional paradigm proposed in the dystopian novel *1984*, raising critical questions about the boundaries between technological efficiency and the preservation of privacy. The system is validated with a simulation using the simulator *Webots* [Michel 1998], in which the UAV must visit a set of targets.

The rest of this paper is organized as follows. Section 2 presents the necessary background. Section 3 presents the proposal. Section 4 shows the experiments executed. Finally, Section 5 presents the conclusion.

2. Background

This section delves into the concepts necessary to understand our proposal. Jason [Bordini and Hübner 2006] is an MAS programming language that implements AgentSpeak (L) [Rao 1996] for BDI agents [Bratman et al. 1988]. In this architecture, **belief** represents the agent's information about the environment, **desire** defines the world states the agent aims to achieve, and **intention** refers to the sequence of actions planned to achieve its objectives. Consequently, agents operate in reasoning cycles, making decisions based on their beliefs and intentions, which are expressed as logical predicates.

Jason agents can interact with the environment and exchange messages with other agents [Hübner et al. 2004]. The Communicator agent extends this communication by enabling the transmission of structured content, such as action, plans and beliefs, through an IoT network [Pantoja et al. 2018]. The ARGO agent, another extension of Jason, uses a Javino middleware [Junger et al. 2016] to communicate directly with the physical environment and its sensory and actuator components. These agents work collaboratively, sharing beliefs and action plans to achieve a common objective.

The UAV model used in this research was the DJI Mavic 2 Pro, whose autonomous operation capabilities are enabled by an integrated sensor suite [Cyberbotics Ltd. 2021]:

- **Navigation**: GPS + magnetometer;
- **Motion Control**: Gyroscope (roll/pitch/yaw) + IMU;
- **Flight Management**: Microcontroller + motor control.

This sensor fusion architecture ensures stable flight operation and precise maneuver execution through continuous feedback loops between hardware components and control algorithms.

For the simulated environment, the open-source *Webots* software (version 2023b) [Michel 1998] was selected, a widely used platform for the development of mobile

robots. This choice was motivated by its accessibility to 3D models, high graphical performance, and its reliance on the *Open Dynamics Engine* (ODE) [Cyberbotics Ltd. 2023] for realistic simulation of physics, collision detection, and friction modeling. To enable the agent to perceive the virtual environment through the controller sensors, a serial communication layer [Freitas et al. 2025] was implemented. This layer facilitates the exchange of communications between the emulated microcontroller port (e.g., ttyEmulatedPort0) and the external microprocessor (e.g., ttyExogenous0) using established protocols.

In this study, the term *Path* refers to a predefined (and potentially randomized) sequence of commands guiding a UAV to waypoints in a 3D Cartesian coordinate system (X, Y, Z), where: X: controls longitudinal movement (forward/backward); Y: governs lateral movement (left/right); Z: manages vertical movement (up/down).

3. Implementation of an Embedded MAS in an Unmanned Aerial Vehicle

This section presents a simulated security system architecture where an embedded Multi-Agent System (MAS) controls UAV operations. The experiment involves two MAS agents and a 3D UAV model in *Webots*, equipped with motors and an integrated GPS, to demonstrate how a MAS can command a UAV and how a security system can act as a central decision-maker (master) for UAV coordination.

The first MAS agent, called *Drone*, directly controls the UAV, while the second agent, Security Secretary, issues high-level commands to the Drone. This architecture can be easily scaled to support multiple Drone agents controlling multiple UAVs—either by adding exogenous ports in *Webots* or implementing additional communication channels in real-world scenarios.

3.1. Technical Implementation and Challenges

This section details the internal structure and operational logic of the UAV implemented in the experiment. Within the *Webots* simulation environment, each robot is represented by a `Robot` node, controlled through an external controller script written in programming languages such as C++, Java, or Python. While functionally similar to an Arduino microcontroller in physical systems, this controller interacts with the simulated environment instead.

The experimental UAV employs a C-based controller that manages its behavior through nine distinct operational states: i) **LANDED**: The UAV is on the ground; ii) **UP**: Ascending vertically; iii) **DOWN**: Descending vertically; iv) **MOVE RIGHT**: Moving to the right (from the UAV's perspective). v) **MOVE LEFT**: Moving to the left; vi) **MOVE FORWARD**: Moving forward; vii) **MOVE BACKWARD**: Moving backward; viii) **HOVERING**: Stabilized in place; ix) **LANDING**: Descending to landing.

Each of these states is triggered by one of the following commands: **hover**: Stabilizes the UAV; **up**: Increases thrust to ascend; **down**: Decreases thrust to descend; **right**: Rolls the UAV to the right; **left**: Rolls the UAV to the left; **forward**: Adjusts pitch to move forward; **backward**: Adjusts pitch to move backward; **land**: Begins landing procedure; **off**: Turns off all motors.

The UAV robot features four motors and an integrated GPS module, as illustrated in Figure 1. It waits for the commands listed below and changes between those states when it receives a command.

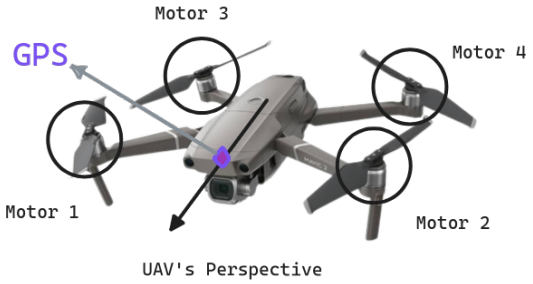

Figure 1. UAV Design

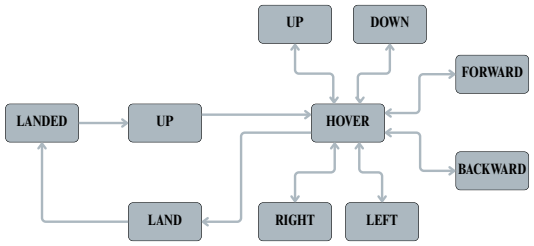

Figure 2. Unmanned Aerial Vehicle State Diagram

To maintain flight stability and perform its operations, the UAV utilizes the world's timestamp along with real-time roll and pitch rates to maintain stability. A proportional-derivative (PD) stabilization algorithm is applied, as shown in the Code 1. During each iteration of the control loop, the UAV updates the speed of each motor accordingly, executing actions based on the combination of `thrust_correction`, `pitch_correction`, and `roll_correction` that are modified when a command is given.

```
roll_correction  += KP_STABILIZE * (-roll)  + KD_STABILIZE * (-roll_rate);
pitch_correction += KP_STABILIZE * (-pitch) + KD_STABILIZE * (-pitch_rate);

motor_speeds[0] = BASE_THRUST + thrust_correction - pitch_correction + roll_correction;
motor_speeds[1] = BASE_THRUST + thrust_correction - pitch_correction - roll_correction;
motor_speeds[2] = BASE_THRUST + thrust_correction + pitch_correction + roll_correction;
motor_speeds[3] = BASE_THRUST + thrust_correction + pitch_correction - roll_correction;
```

Code 1. Stabilization algorithm.

These constants and equations determine how the UAV reacts to commands and ensure its flight remains stable under various scenarios. However, it is not sufficient to maintain stability in the air when being controlled by a MAS. This is because there is a latency between the MAS sending a command and the robot executing it. So, to maintain stability, the Robot turns itself back to the **HOVERING** state for stabilizing between actions, as Figure 2 suggests.

3.2. The Drone to Unmanned Aerial Vehicle

The Drone MAS has an ARGO agent (named *eye*) and a Communicator agent (named *teletela*) as the Figure 3 describes. The *eye* agent has the function to command the UAV robot, sending commands through an *act* from the ARGO agent, where each command is named after the states in Figure 2. When the *eye* receives a *Path*, it executes *Path* instructions by sending commands to the UAV Robot and stops sending a specific command when the UAV reaches a specified location in GPS.

This command communication is possible because of a simulated port created to connect to the UAV Robot. In this way, the *eye* agent connects the *Drone* MAS to *Webots* using the **/dev/ttyExogenous0** port, an approach detailed by [Freitas et al. 2025] and shown in Figure 4.

On the other hand, *teletela* has the function to ask the Security Secretary for the path, communicates when a path is done to the Security Secretary, and informs *eye* the Path given by the Security Secretary.

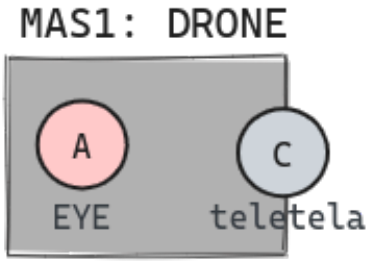

Figure 3. ARGO Agent- Drone

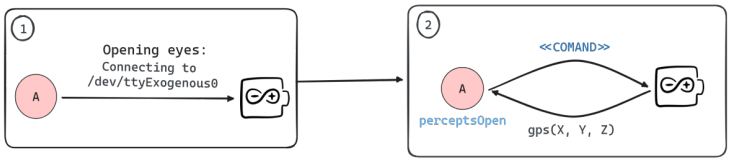

Figure 4. Utilizing Exogenous Port to Open The Big Agent Eyes

3.3. Communication between MAS

The *teletela* agent requests a *Path* (PathRequest) from the Security Secretary agent when it believes that the *eye* agent is ready, and then waits to receive it. After receiving the plan, the communicator agent is responsible for instructing the ARGO agent *eye* with this action plan and expects it to execute the mission.

In the context of this research, the communicator agent Security Secretary, which holds Paths to be executed, is responsible for determining, at a certain time interval, a path with the corresponding action plans for the communicator agent *teletela*.

This communication is made possible through ContextNet technology [Endler et al. 2011], an IoT middleware that enables communication between agents based on their respective UUIDs [Leach et al. 2005] as belief bases and a shared address, which in this research is `skynet.chon.group`, as illustrated in Figure 5.

In this way, agents can exchange messages and illocutionary forces such as *tell-How* and *askHow* to, respectively, communicate and request new action plans from each other. This can be done either by targeting specific agents using their UUIDs or simultaneously through a server via *ConnectCN*, and a logical port, enabling communication on a larger scale [Lazarin et al. 2021].

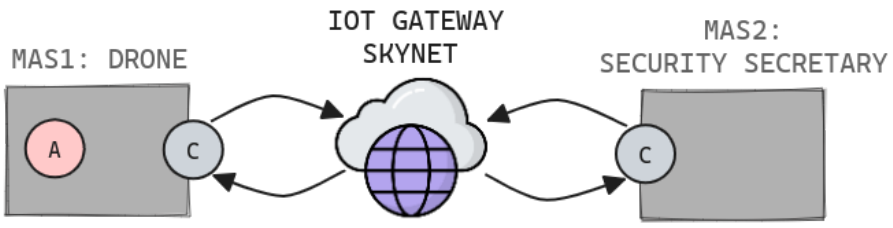

Figure 5. Multi-agent System communication over *ChonNet* [Lazarin et al. 2023].

4. Experiments

Summarizing, the *eye* agent connects to the robot body. Then it speaks to the *teletela* agent when it is ready. The *teletela* agent asks the Security Secretary for a *Path*, which the Security Secretary then sends back to the *teletela* agent. Consequently, *teletela* tells the *eye* the *Path* and commands it to achieve it. When the *eye* agent starts the *Path*, it commands the Robot UAV to execute its *Path*, and the robot controls its motors.

The experimental process takes place in a simulated environment in Webots, which consists of a house - which the embedded UAV will navigate - surrounded by trees, gravity effects, a box, a street, a windmill, and a car. All objects, except the house, are merely decorative, while the house serves as a reference point for the drone's navigation.

In this experiment, Secretary MAS sent the following Path to the Drone. This Path contains 8 points in the world's Cartesian coordinate system formatted as a Path: `!takeoff; !up(20.0); !right(15.0); !forward(-45.0); !left(-1.0); !backward(0.0); !right(-1.0); !backward(1.0); !down(2.0); !land; !turnOff; !contactBack.`. The numerical values associated with each command indicate a point with which the Drone should continue executing the corresponding action to complete the Path. These values can be interpreted as: Continue going to value - Continue going forward to -45.0 -. This Path is in Secretary's mind and is passed to Teletela upon request.

To expand this experiment in a real-world scenario, additional implementations are required, such as adjusting configuration variables to match the physical characteristics of UAVs, developing turning states, turning methods, and plans for the *eye* agent. This is necessary to create simpler paths. For example, if a *Path* contains 'forward(3.0)'—go forward to 3.0 in the X axis— and the UAV is on -3.0 in the X axis, it has already surpassed 3.0 and will never stop moving forward. This is an example of a wrong command made by the Security Secretary MAS.

This example illustrates how important it is to truly have a correct *Path* and the importance of a simulation to make sure the *Path* is correct. The *Drone* MAS does not possess reasoning capabilities. It merely follows the orders issued by the Security Secretary, MAS. In a real-world scenario, to preserve UAVs' operational safety, they should be equipped with more adaptive behaviors, such as stopping when encountering obstacles and interpreting paths in a more adaptive way.

4.1. Reproducibility

In order to ensure the reproducibility of this work, the source code, the simulator implementation and a video demonstration are available[1].

5. Conclusion

This study successfully demonstrates remote UAV control using an embedded Multi-Agent System (MAS). The proposed architecture integrates intelligent agents with physical hardware components, coordinating randomized routes in real time through simultaneous communication. By employing an ARGO agent to bridge the physical microprocessor with JASON communication extensions, we prove the feasibility of distributed UAV control that's both functional and cost-effective.

The results indicate strong real-world applicability, though key challenges remain: improving environmental recognition capabilities, standardizing event detection protocols, and developing dynamic state-update mechanisms. These advancements would not only enhance the technology's efficiency but also spark crucial debates about implementing intelligent drones in society without undermining core ethical values.

[1] `https://papers.chon.group/WESAAC/2025/BigAgent/`

References

AL-Dosari, K. and Fetais, N. (2023). A New Shift in Implementing Unmanned Aerial Vehicles (UAVs) in the Safety and Security of Smart Cities: A Systematic Literature Review. *Safety*, 9(3). DOI: 10.3390/safety9030064.

Bordini, R. H. and Hübner, J. F. (2006). BDI Agent Programming in AgentSpeak Using Jason. In *Computational Logic in Multi-Agent Systems*, volume 3900, pages 143–164. Springer Berlin Heidelberg, Berlin, Heidelberg. DOI: 10.1007/11750734_9.

Bratman, M. E., Israel, D. J., and Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(3):349–355. DOI: 10.1111/j.1467-8640.1988.tb00284.x.

CNN Brasil (2025). Atlas: 9 em 10 brasileiros consideram criminalidade uma grande preocupação. URL: https://www.cnnbrasil.com.br/politica/atlas-9-em-10-brasileiros-consideram-criminalidade-uma-grande-preocupacao/ Accessed: 08-04-25.

Cyberbotics Ltd. (2021). *Webots Documentation: Mavic 2 Pro*. Cyberbotics Ltd. `https://www.cyberbotics.com/doc/guide/mavic-2-pro?version=R2021a`.

Cyberbotics Ltd. (2023). *ODE (Open Dynamics Engine) Reference Manual*. Cyberbotics Ltd. URL: `https://cyberbotics.com/doc/reference/ode-open-dynamics-engine` Accessed: 06-29-25.

Endler, M., Baptista, G., Silva, L. D., Vasconcelos, R., Malcher, M., Pantoja, V., Pinheiro, V., and Viterbo, J. (2011). ContextNet: context reasoning and sharing middleware for large-scale pervasive collaboration and social networking. In *Proceedings of the Workshop on Posters and Demos Track*, PDT '11, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/2088960.2088962.

Freitas, B., Lazarin, N., Pantoja, C., and Viterbo, J. (2025). Integrating Simulated Exogenous Environments to Support the Learning Process of the Embedded MAS Approach. In *Anais do XXXIII Workshop sobre Educação em Computação*, pages 1195–1206, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wei.2025.9115.

Hübner, J. F., Bordini, R. H., and Vieira, R. (2004). Introdução ao desenvolvimento de sistemas multiagentes com Jason. In *Anais da XII Escola de Informática da SBC*, volume 2, pages 51–89. URL: `https://scholar.google.com.br/scholar?cluster=5089382793799834882`.

Jesus, V., Manoel, F., and Pantoja, C. (2019). Protocolo de Interação Entre SMA Embarcados Bio-Inspirado na Relação de Predatismo. In *Anais do XIII Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações*, pages 95–106, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wesaac.2019.33343.

Junger, D., Silva, J., and Pantoja, C. (2016). An Analysis of Javino Middleware for Robotic Platforms Using Jason and JADE Frameworks. In *Anais do X Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações*, pages 163–168, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wesaac.2016.33217.

Lazarin, N., Pantoja, C., and Jesus, V. (2021). Um Protocolo para Comunicação entre Sistemas Multi-Agentes Embarcados. In *Anais do XV Workshop-Escola de Sistemas*

de Agentes, seus Ambientes e Aplicações, pages 157–168, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wesaac.2021.33416.

Lazarin, N., Pantoja, C., and Viterbo, J. (2023). Towards a Toolkit for Teaching AI Supported by Robotic-agents: Proposal and First Impressions. In *Anais do XXXI Workshop sobre Educação em Computação*, pages 20–29, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/wei.2023.229753.

Leach, P. J., Salz, R., and Mealling, M. H. (2005). A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122. DOI: 10.17487/RFC4122.

Michel, O. (1998). Webots: Symbiosis Between Virtual and Real Mobile Robots. In Heudin, J.-C., editor, *Virtual Worlds*, pages 254–263, Berlin, Heidelberg. Springer Berlin Heidelberg. DOI: 10.1007/3-540-68686-X_24.

Osmani, K. and Schulz, D. (2024). Comprehensive Investigation of Unmanned Aerial Vehicles (UAVs): An In-Depth Analysis of Avionics Systems. *Sensors*, 24(10). DOI: 10.3390/s24103064.

Pantoja, C., Soares, H. D., Viterbo, J., and Seghrouchni, A. E. F. (2018). An Architecture for the Development of Ambient Intelligence Systems Managed by Embedded Agents. pages 215–249. DOI: 10.18293/SEKE2018-110.

Pantoja, C. E., Jesus, V. S. d., Lazarin, N. M., and Viterbo, J. (2023). A Spin-off Version of Jason for IoT and Embedded Multi-Agent Systems. In *Intelligent Systems*, pages 382–396, Cham. Springer Nature Switzerland. DOI: 10.1007/978-3-031-45368-7_25.

Pantoja, C. E., Stabile, M. F., Lazarin, N. M., and Sichman, J. S. (2016). ARGO: An Extended Jason Architecture that Facilitates Embedded Robotic Agents Programming. In *Engineering Multi-Agent Systems*, pages 136–155, Cham. Springer. DOI: 10.1007/978-3-319-50983-9_8.

Petritoli, E., Leccese, F., and Ciani, L. (2018). Reliability and maintenance analysis of unmanned aerial vehicles. *Sensors*, 18(9):3171. DOI: 10.3390/s18093171.

Rao, A. S. (1996). Agentspeak(l): Bdi agents speak out in a logical computable language. In *Agents Breaking Away*, pages 42–55, Berlin, Heidelberg. Springer Berlin Heidelberg. DOI: 10.1007/BFb0031845.

SSP-RJ (2025). Segurança em Números 2025 (ano-base 2024). URL: `https://www.rj.gov.br/isp/node/1507`.

World Bank (2016). Brazil - Systematic country diagnostic : retaking the path to inclusion, growth and sustainability. Technical Report 101431-BR, World Bank, Washington, DC. URL: http://documents.worldbank.org/curated/en/284561467996687489.

Younes, N., Boukhdir, K., Moutaouakkil, F., and Medromi, H. (2016). A multi-task distributed vision system embedded on a hex-rotorcraft uav. *International Journal of Advanced Computer Science and Applications*, 7(10). DOI: 10.14569/IJACSA.2016.071016.