UDC 004.042

A. Akhaladze, O. Lisovychenko

ARTIFICIAL INTELLIGENCE-BASED DECENTRALIZED CONTROL OF A HETEROGENEOUS UNMANNED AERIAL VEHICLE SWARM UNDER INTERMITTENT COMMUNICATION CONDITIONS

Abstract. This paper addresses the problem of partially successful or failed mission execution by multiple drones operated centrally by human pilots over an unreliable control channel. We propose an artificial-intelligence-based approach to the decentralized control of a heterogeneous swarm of unmanned aerial vehicles (UAVs) under intermittent communication. Swarm heterogeneity—stemming from UAVs with diverse sensing, mobility, and endurance capabilities—complicates coordination, while communication outages demand a high degree of on-board autonomy. The method relies on reinforcement-learning techniques that enable individual UAVs to make decisions locally and to adapt to changes in the environment and in swarm composition. The approach improves the resilience, efficiency, and fault tolerance of the system, allowing the swarm to accomplish complex tasks such as reconnaissance, environmental monitoring, and search-and-rescue operations without dependence on a centralized control node. Emphasis is placed on the design of algorithms that ensure effective interaction and cooperative task execution even in the presence of partial or complete loss of inter-UAV communication or the failure of individual agents.

Introduction

Modern military and civilian missions increasingly rely on unmanned aerial vehicles (UAVs). The swarm-based concept—deploying many UAVs to accomplish spatially distributed tasks—can outperform single-vehicle solutions. Classical centralized control, however, in which a single operator or command center issues direct commands to every drone, is critically vulnerable: failure of the central node or loss of the control link collapses the entire system.

The problem is exacerbated under adversarial conditions (e.g., electronic-warfare jamming) or in challenging terrain such as urban canyons or mountainous regions, where a stable link cannot be guaranteed. Swarm heterogeneity—UAVs with different payloads (optical sensors, relays, effectors), flight characteristics, and onboard computing power—adds another layer of complexity.

The goal of this research is to improve mission performance by enhancing intra-swarm interaction inside a heterogeneous UAV swarm with decentralized control that

leverages artificial-intelligence methods to guarantee autonomy and adaptability under unreliable communications.

Rule-based swarm algorithms (flight formations, leader–follower schemes, etc.) [1] perform well under ideal conditions but lack flexibility when tasks change or vehicles are lost. Ground-control stations such as QGroundControl [6] are powerful for mission planning and monitoring, yet their role must shift in a decentralized architecture.

Problem statement

We aim to replace direct per-drone teleoperation with an architectural and algorithmic framework that enables every swarm member to make locally optimal decisions that serve the collective objective [2]. To boost mission effectiveness in a heterogeneous swarm operating over an intermittent channel, the following tasks must be solved:

1. propose a modular architecture that embeds a decision-making mechanism onboard each UAV;
2. design a control-decision algorithm [3] that, based on position data received from neighbors within communication range, drives each drone toward the global mission goal;
3. introduce a reward–penalty scheme that uses AI techniques to evaluate each drone’s decision with respect to the swarm’s objective.

Proposed solution

We adopt a hybrid architecture in which every UAV carries an ArduPilot flight controller (FC) and a companion computer (CC) based on Raspberry Pi or NVIDIA Jetson. The FC ensures low-level stabilization and executes primitive commands (loiter, waypoint navigation) sent via MAVLink—acting as the spinal cord. The CC (e.g., Jetson Nano [7]) functions as the “brain”: it runs AI logic, processes sensor data, exchanges information with peers, and issues high-level commands to the FC through MAVLink [8]. Running Linux and Robot Operating System (ROS) [9] on the CC yields a flexible modular stack that reuses existing packages:

1. MAVROS [10]—a ROS/MAVLink bridge;
2. OpenCV [11]—computer-vision algorithms;
3. SLAM/VIO toolkits [12]—GPS-denied navigation and mapping;

Using off-the-shelf packages makes it possible to deploy the decision-making logic onboard every drone in the swarm and to shorten development time by re-using existing algorithms and part of the mathematical machinery. The AI module contains two principal classes of ROS nodes.

1. MAVROS node is connected to the flight controller (FC) via a UART interface. It subscribes to telemetry streams from the FC—GPS fix, altitude, battery state—and

 ISSN 1560-8956

republishes them as ROS topics accessible to the other nodes. Conversely, it listens to high-level commands from peer ROS nodes (e.g., a desired 3-D set-point) and forwards them to the FC through the MAVLink protocol.

2. AI node (e.g., swarm_logic_node, vision_node) perform all computationally intensive tasks: they ingest the telemetry topics provided by MAVROS, analyse the data, exchange information with neighbour drones, and publish high-level control directives (updated target way-points) to the topics monitored by MAVROS. This separation allows each UAV to reason autonomously while maintaining seamless integration with the low-level flight stack.

The decision-making logic relies on reinforcement learning (RL) [4] executed by an RL agent; in our case we employ its multi-agent extension—multi-agent reinforcement learning (MARL) [5].

Problem formulation for a UAV RL agent (see fig. 1).

1. State S — a vector comprising the UAV’s own data (coordinates, velocity, battery level, payload type), information received from neighbouring drones (their states and intents), and a descriptor of the current task.

2. Action A — selection of the next tactical manoeuvre, e.g. “move to reconnaissance waypoint X,” “start scanning,” “relay data from neighbour Y,” “take a covering position.”

3. Reward R — a scalar score assigned to the action. The reward, which drives the emergence of cooperative behaviour, can include several components:

a. + bonus for accomplishing part of the global mission (e.g., a new sector successfully scanned);
b. + bonus for maintaining a data link with neighbouring agents, which encourages the formation of a resilient mesh network;
c. - small constant penalty for excessive energy use, driving the drone to complete the task quickly and efficiently rather than loiter aimlessly;
d. - large penalties for hazardous actions such as potential collisions or exiting the authorised flight zone;

The total reward that the agent receives for each step or synthesized and executed solution is the sum of all components and is given by the formula (1):

$$R_{total} = R_{task} + R_{comm} + R_{energy} + R_{risk}, \quad (1)$$

where:

R_{task} - is the primary positive incentive that motivates the agent to perform useful work;

R_{comm} - rewards inter-agent connectivity, encouraging the swarm to remain cohesively linked;

R_{energy} - is a constant, low-magnitude penalty promoting energy-efficient flight;

R_{risk} - comprises high-magnitude penalties for unsafe actions, designed to dominate any potential gain from risky behaviour.

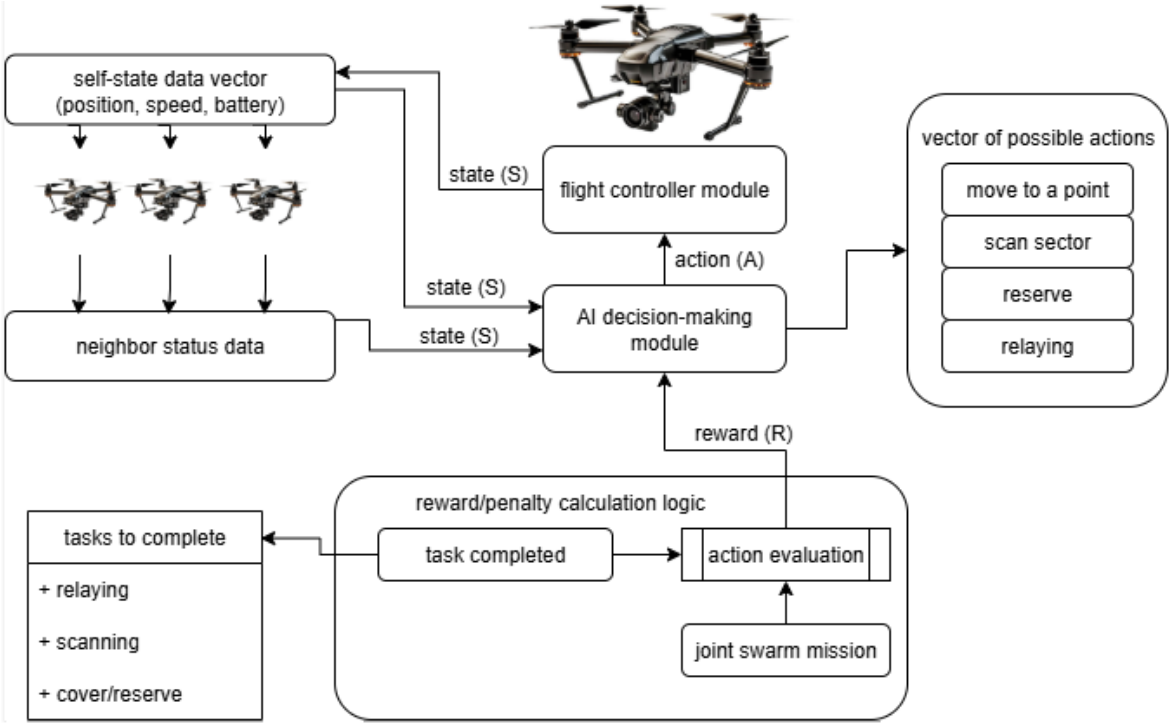

Figure 1. Scheme of the solution synthesis mechanism by each swarm member

Let's set the general goal of the swarm - exploration by scanning a new territory divided into sectors for the presence of objects. Then we will get the following logic for calculating rewards and penalties:

$$R_{task} \Rightarrow \left[\frac{+100 \text{ for a successfully scanned new sector}}{-5 \text{ for re-checking what has already been scanned}}\right] \quad (2)$$

$$R_{comm} \Rightarrow [3 * n], \quad (3)$$

where:

n - number of neighboring agents within the communication network range.

$$R_{energy} \Rightarrow [-1 * \text{throttle}^2], \quad (4)$$

where:

throttle - negative component of the ineffective acceleration assessment.

$$R_{risk} \Rightarrow \left[\frac{-200 \text{ when approaching a neighbor at a dangerous distance (1m)}}{-300 \text{ when leaving the designated mission area}}\right] \quad (5)$$

Each action of each agent in the swarm is evaluated by a signed integer value. A score with a negative value is evaluated as undesirable, and its value determines the criticality of the action for achieving the goal of the swarm. A positive score of the action of an individual member of the swarm is evaluated as an action directed towards the fulfillment of the general goal of the swarm, and its numerical value determines the importance of an

 ISSN 1560-8956

individual action in the plane of achieving a given goal. Thanks to such a system of action scores, each UAV is a component of the general solution of the problem of setting the maximum score for the actions of each agent in the swarm, where the maximum value is possible when each UAV acts not only in its own interests, but also for the benefit of the entire swarm, since collective success leads to the maximum individual reward. One of the modern MARL algorithms can be used for training, for example, Multi-Agent Proximal Policy Optimization (MAPPO) [13].

Simulation and experimental results

To validate the proposed approach, a simulation environment was developed using Gazebo [14] and software-in-the-Loop (SITL) [15], which allows emulating flight physics and the operation of ArduPilot [16].

During the simulation, a heterogeneous swarm (5 reconnaissance drones, 2 repeater drones) was tasked with conducting reconnaissance of the territory.

The results showed:

High adaptability: when communication between reconnaissance drones is lost, the repeater drone autonomously occupies a position between them to restore the communication link.

Fault tolerance: when simulating the failure of one of the reconnaissance drones, the other UAV, without receiving data from the specified agent, independently redistributes the task and continues reconnaissance of free sectors.

Efficiency: The decentralized approach demonstrated a more efficient way to accomplish the overall shared goal of a heterogeneous UAV swarm in an unstable communication environment compared to a centralized model that constantly waited to reestablish communication with the command center.

Analysis of results

The proposed modular architecture using a separate module for performing tasks of making its own decision on the side of each swarm member allows building a swarm capable of fulfilling a common goal in an unstable communication environment.

The implemented algorithm for making a decision on the side of each swarm member, by analyzing data on the position of neighbors, allows for the implementation of complex autonomous behavioral patterns in the absence of communication with the operator or ground station.

The developed approach for evaluating decisions made by calculating penalties and rewards allows for the generation of individual autonomous decisions on the side of each swarm member, aimed at achieving the general goal of the swarm in the presence of unstable

communication with the operator or ground station or in the complete absence of external control influence.

The implementation of the tasks allowed for a more efficient implementation of the set goal by combining them into a heterogeneous swarm of drones operating in conditions of limited or completely absent communication channel with the central control node.

References

1. Akhaladze A.E. Using IoT to synchronize flight trajectories of drones // Adaptive automatic control systems. 2021. No. 39. C.20-26 URL: http://asac.kpi.ua/article/view/247381, https://doi.org/10.20535/1560-8956.39.2021.247381

2. Akhaladze A.E. Synchronization of flight trajectories based on the "Internet of Things" architecture when implementing swarm control Adaptive automatic control systems. 2022. No. 40. C. URL: http://asac.kpi.ua/article/view/261536, https://doi.org/10.20535/1560-8956.40.2022.261536

3. Liu, S., Zou, C., & Song, Y. (2019). Multi-objective optimization of UAV path planning based on genetic algorithm. Complexity, 2019, 1-15 URL: https://www.researchgate.net/publication/322920720_Multi-objective_genetic_algorithm_for_civil_UAV_path_planning_using_3G_communication_networks

4. Changxi Zhu, Mehdi Dastani, Shihan Wang. A survey of multi-agent deep reinforcement learning with communication. // Autonomous Agents and Multi-Agent Systems. 2024. 38:4 URL: https://doi.org/10.1007/s10458-023-09633-6

5. Yang Y., Ma C., Ding Z., McAleer S., Jin C., Wang J. Game-Theoretic Multiagent Reinforcement Learning // arXiv preprint arXiv:2011.00583. 2020. URL: https://arxiv.org/abs/2011.00583

6. Agar, D., Grubba, G., & contributors. (2023). QGroundControl (Version 4.2.0) [Computer software]. Retrieved July 25, 2025, from https://github.com/mavlink/qgroundcontrol

7. NVIDIA Corporation. (2025). Jetson Nano developer kit: Technical specifications [Electronic resource]. Retrieved July 25, 2025, from https://developer.nvidia.com/embedded/jetson-nano-developer-kit

8. Mason, L., Wolf, L., & contributors. (2024). MAVLink: Micro air vehicle communication protocol (Version 2.0) [Electronic resource]. Retrieved July 25, 2025, from https://mavlink.io/en/

9. Open Source Robotics Foundation. (2023). Robot Operating System (ROS) (Humble Hawksbill) [Computer software]. Retrieved July 25, 2025, from https://www.ros.org/

10. Mei, L., Takasu, T., & contributors. (2024). MAVROS: MAVLink extendable communication node for ROS (Version 1.16.0) [Computer software]. Retrieved July 25, 2025, from https://github.com/mavlink/mavros

11. Bradski, G., Pisarevsky, V., & contributors. (2024). OpenCV: Open-source computer vision library (Version 4.9.0) [Computer software]. Retrieved July 25, 2025, from https://opencv.org/

12. Qin, T., Li, P., & Shen, S. (2023). VINS-Fusion: Robust visual–inertial navigation system (Version 1.0.0) [Computer software]. Retrieved July 25, 2025, from https://github.com/HKUST-Aerial-Robotics/VINS-Fusion

13. Yu, C., Velu, A., Vinitsky, E., et al. (2021). MAPPO: Multi-agent proximal policy optimization (arXiv preprint arXiv:2103.01955). Retrieved July 25, 2025, from https://arxiv.org/abs/2103.01955

14. Open Source Robotics Foundation. (2023). Gazebo: 3-D robotics simulator (Version 11.15.0) [Computer software]. Retrieved July 25, 2025, from https://gazebosim.org/

15. ArduPilot Development Team. (2024). SITL (Software-in-the-Loop) simulator (Version 4.2) [Computer software]. Retrieved July 25, 2025, from https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html

16. ArduPilot Development Team. (2024). ArduPilot: Open-source autopilot software (Version 4.4.0) [Computer software]. Retrieved July 25, 2025, from https://ardupilot.org/