IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 March 2024, accepted 20 May 2024, date of publication 27 May 2024, date of current version 7 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3406133

RESEARCH ARTICLE

Learning-in-Games Approach for the Mission Planning of Autonomous Multi-Drone Spatio-Temporal Sensing

VITTORIO U. CASTRILLO[1], DOMENICO PASCARELLA[1], GIANPAOLO PIGLIASCO[1], IVAN IUDICE[1], AND ANGELA VOZELLA[2]

[1]Security of Systems and Infrastructures Laboratory, Italian Aerospace Research Centre (CIRA), 81043 Capua, Italy
[2]Reliability Availability Maintainability Safety and Security Department, Italian Aerospace Research Centre (CIRA), 81043 Capua, Italy

Corresponding author: Vittorio U. Castrillo (v.castrillo@cira.it)

This work was supported by the Italian National Project "Maturazione Tecnologie Innovative Mini e Micro Droni (MATIM)" through Programma Nazionale di Ricerche Aerospaziali (PRORA) under Grant DM662.

ABSTRACT Mission planners are one of the major classes of autonomy software and their design is especially challenging in the case cooperation autonomy is required for unmanned multi-vehicle systems. A clear example of this is given by the applications of teams of drones, such as multi-drone spatio-temporal sensing. Here, drone teams act as mobile and cooperative sensor networks to simultaneously collect sensor data in areas of interest and to allow detailed computation on the sensed data. For the design of cooperative and autonomous drone teams, mission planning shall be accomplished in the form of coordinated sensing to optimally assign the different sensing tasks and routes to each drone, employing task allocation and route planning as the basic pillars to maximize the multi-drone mission effectiveness. This work proposes a dynamic and decentralized mission planner for a drone team performing autonomous and cooperative spatio-temporal sensing. The design exploits the learning-in-games framework for the processing of optimal routes in reasonable time frames. Two ad-hoc variants of the binary log-linear learning are proposed as a coordination algorithm to manage both task allocation and route planning, by demonstrating reachability and reversibility properties. Also, the work describes an experimental analysis of the proposed solutions by means of model-in-the-loop simulations, in order to provide a preliminary tune of the main learning parameters for both solutions.

INDEX TERMS Multi-drone systems, multi-drone sensing, learning in games, multi-agent systems.

NOMENCLATURE

$\mathbb{B}_i^{(m)}(\mathcal{M}_i)$	= Set of all the ascendant plans of $\mathcal{M}_i$ with orders until m.
$\mathbb{C}_i^{w}(\mathcal{M}_i)$	= Constrained set built starting from the mission plan $\mathcal{M}_i$ and considering a planning window w.
$\mathbb{D}$	= Wet of drones.
$\mathbb{F}_i^{(m)}(\mathcal{M}_i)$	= Wet of all the descendant plans of $\mathcal{M}_i$ with orders until m.
$\mathbb{M}_i$	= Set of admissible mission plans for the ith drone.
$\mathbb{M}_i^{[+j]}(\mathcal{M}_i)$	= Set of all the descendant plans of order j for the mission plan $\mathcal{M}_i$.
$\mathbb{W}$	= Region of interest.
$\mathbb{T}$	= Set of sensing tasks.
$C_{\mathrm{E},n}$	= Occurrence cost of the reference event for the waypoint $\mathbf{wp}_n$.
ECI	= Expected Cost of Ignorance.
G	= Global function (potential function).
$\mathcal{M}$	= Joint mission plan.
$\mathcal{M}_i$	= Mission plan for the ith drone (or agent).
$\mathcal{M}_i^{[+m]}$	= mth order descendant of $\mathcal{M}_i$.
$\mathcal{M}_i^{[-m]}$	= mth order ascendant of $\mathcal{M}_i$.

The associate editor coordinating the review of this manuscript and approving it for publication was Vishal Srivastava.

$\mathcal{M}_i^{\emptyset}$ = Joint mission plan $\mathcal{M}$ with no action taken by the ith drone.
$\mathcal{M}_{-i}$ = Set of the individual plans of the drones different from i in the team $\mathbb{D}$.
$\mathcal{M}^{\text{opt}}$ = Optimal joint mission plan.
N_{d} = Number of drones.
N_{t} = Number of sensing tasks.
$P_{\text{E},n}$ = Occurrence probability of the reference event for the waypoint $\mathbf{wp}_n$.
J = Mission duration.
ceil(x) = The smallest integer greater than or equal to the real number x.
$\mathbf{d}_i$ = ith drone.
i = ith agent.
p_n = Priority of the nth sensing task.
$\mathbf{t}_n$ = nth Sensing task.
$\mathbf{t}_{i_n}$ = nth Sensing task assigned to i in $\mathcal{M}_i$.
u_i = Utility function for the ith drone.
v_i = Cruise speed of the ith drone.
w = planning window.
$\mathbf{wp}_n$ = Waypoint of the nth sensing task

I. INTRODUCTION

Autonomy is defined as ''an unmanned system's own ability of sensing, perceiving, analysing, communicating, planning, decision-making, and acting, to achieve its goals as assigned by its human operator(s) through designed human-robot interface or by another system'' [1]. One of the main classes of autonomy-enabling software regarding Autonomous Vehicles (AVs), and in particular Unmanned Vehicles (UVs), is represented by mission planners [2]. In general, mission planners make decisions about vehicle actions by projecting the future outcomes of these actions according to models of the vehicle's behaviour and environment, and evaluating the desirability of the outcomes according to an evaluation function. The evaluation function may encode the vehicle's goals and constraints, returning high utilities for plans that meet the mission goals without violating the constraints. The planning technology necessarily encompasses some form of search through an action space, which is commonly an NPhard problem [2]. At the mission planning level, the following functions are usually performed [3]: (i) the selection and ordering of a subset of mission waypoints that make up the plan; (ii) the definition of the trajectories between the mission waypoints at a low level of resolution. Thus, a mission planner system usually focuses on the specific issues of: task allocation, in order to assign mission waypoints (targets) to the AV in a specified sequence; route planning, in order to determine the optimal preliminary trajectories for the AV.

Moreover, the research and design activities of UVs are facing a major transformation in the past two decades [4], [5]. While previously the focus was on developing large vehicles capable of carrying significant payloads over large distances, the recent technological improvements have shifted research and industrial interests to cooperative and autonomous operations involving multiple UVs [6], forming full-fledged multi-UV systems and achieving common objectives that may be dynamically changed during the mission execution. This implies the adoption of cooperative-autonomy paradigms for multi-UV mission planners.

In general, cooperative autonomy studies how autonomous agents (i.e., separate decision-making entities) should work together to achieve common goals [7]. Thus, a Cooperative Autonomous System (CAS) is a system engineered as a collection of autonomous agents, which shall accomplish common goals. In the domain of distributed intelligence, cooperation is a form of interaction in which: entities are aware of other entities; entities share goals; entities' actions are beneficial to their teammates [8]. In the domain of multirobot and multi-UV systems, cooperation represents joint operations or actions in a group of robots systematized in the form of a Multi-Agent System (MAS) [9], which is also named Multi-Agent Robot System (MARS). In such cooperation, robots pay attention to their own work and to the tasks from other partners [9].

One of most significant applications about cooperative multi-UV systems is represented by drones and multi-drone systems. Indeed, if the Region Of Interest (ROI) is large and/or the objectives are several, a standalone-drone mission may take a considerable amount of time and may entail poor performance. For example, in the case of spatio-temporal sensing, the use of standalone drones may reduce the usefulness and the reliability of the collected information. Drone teams may overcome these issues, being a networked set of drones with a common mission, in which all members are assigned specialized and different tasks to accomplish the global mission [10], [11]. Such networked sets of drones may achieve: (i) a group performance that is expected to exceed the sum of the performance of the individual drones; (ii) multiple simultaneous interventions; (iii) an efficient cover of large areas, optimizing the available resources; (iv) fault-tolerant and resilient missions, by providing redundancy and capability of reconfiguration in case of a failure of a drone; (v) costefficiency, since a team of low-cost drones may represent a less expensive solution with respect to the equivalent standalone and heavier drone for the same mission. Moreover, drone teams are gaining greater relevance also in the context of cooperative system-of-systems integrating both manned and unmanned aerial vehicles [12].

For the design of cooperative and autonomous drone teams, task allocation and route planning are the basic pillars to coordinate the overall team and to maximize the multi-drone mission effectiveness, guaranteeing optimal task completion [13]. In the case of spatio-temporal sensing missions, drone teams act as mobile and cooperative sensor network to simultaneously collect sensor data in areas of interest and to allow detailed computation on the sensed data. Here, mission planning shall be accomplished in the form of coordinated sensing to optimally assign the different sensing tasks and routes to each drone, while complying with

the constraints related to sensing requirements and drone features [14].

This work proposes a dynamic and decentralized mission planner for a drone team performing autonomous and cooperative spatio-temporal sensing. The design of the planner allows for an efficient and resilient management of simultaneous sensing operations. It exploits game theory for the assignment of sensing tasks and for the processing of optimal routes in reasonable time frames, laying the foundation for autonomous real-time implementation. Unlike other approaches, the proposed game-theoretic framework is able to manage the team mission planning with multi-priority tasks, by embedding the concept of expected cost of ignorance for a sensing task.

We leverage on some our previous works, which applied Markov games and the Distributed Stochastic Algorithm for the configuration of a heterogeneous drone team in persistent surveillance applications [15]. This work introduces some changes by applying the learning-in-games framework for the coordination of the team. For the learning design, we already proposed an ad-hoc variant of best response and of log-linear learning as a possible learning algorithm for autonomous detection and recognition, analyzing the effectiveness of the approach by comparing it with a centralized solution in preliminary model-in-the-loop testing [16]. We extend this previous work with the following main contributions:

- we implement an action-selection strategy based on a different data-normalization function;
- we provide a mathematical analysis of the reachability and reversibility properties for the designed learning-ingames approach;
- we provide a detailed experimental analysis of the coordination algorithm, to assess the influence of significant learning parameters and to preliminarily tune their values with respect to the environment.

The experimental analysis is performed by means of modelin-the-loop simulations, exploiting the implementation of the mission planner in an Agent-Based Modelling and Simulation (ABMS) environment.

The remainder of this paper is organized as follows. Section II describes the related research, focusing on the topics of multi-agent coordination, multi-drone cooperation, multidrone task allocation, and multi-drone routing. Section III introduces the application scenario and formally states the reference problem. Section IV describes the proposed gametheoretic solution, by presenting the design of the game model and of the learning-in-games method, and the implementation of the coordination algorithm. Section V describes the experimental analysis of the proposed solution. Section VI concludes the work.

II. RELATED WORK

Recently, the decision-making and the control of a group of autonomous systems (or autonomous agents) have been intensively investigated from different perspectives. Specifically, in the case of MARS, the identification of a common framework for developing the best engineering solution is a challenging topic since these systems cannot be studied and evaluated by generalizing the case of a single robot [9]. Currently, MARS with distributed coordination have already demonstrated their capabilities for being robust, adaptive, flexible and scalable [17]. However, some relevant issues exist for task allocation and learning since these require the decision-making process to be distributed in a robust and efficient manner, considering both processing and communication overhead [18].

For the specific case of multi-drone cooperation, several architectures have been proposed [19]. Relevant research has been performed so far to advance the applications of multidrone systems, especially in regard to the analysis of drone communication aspects [20], [21], [22]. Nevertheless, the mission planning process and the related problems, such as multi-drone task allocation and routing, have become a challenging issue for the achievement of real autonomy in drone operations for different domains [23], [24]. This applies in particular for multi-drone spatio-temporal sensing [14].

In regard to multi-drone task allocation, references [25] and [26] provides a detailed survey of the most recent decentralized algorithms. Apart from the distinction in centralized and decentralized approaches, these algorithms may be divided in the following taxonomy presented in [26].

- Optimization-based algorithms - These methods adapt classical optimization methods to solve the task allocation problem. They may be deterministic or stochastic. An example of deterministic Hungarianbased method is represented by the Multi-UAV Collaborative Target Allocation (MCTA) algorithm [27]. Stochastic algorithms include bio-inspired methods, which mimic specific biological behaviors to deal with the optimization problem related to task allocation [25].
- Auction-based algorithms - These methods rely on economic principles, designing the agents by means of negotiation protocols to bid on tasks in an auction. They may be centralized or decentralized. Some examples are represented by the Consensus-Based Bundle Algorithm (CBBA) [28] and by the Contract Net Protocol (CNP) [29].
- Game-theoretic algorithms - These methods apply frameworks based on game theory to design the interaction strategies amongst the agents, which are assumed to behave as independent and self-interested players, or as group of these players. Here, the general objective is to find an equilibrium, such as a Nash equilibrium, between the decisions deliberated by the agents [30].
- Hybrid algorithms - These methods represent a combination of the aforementioned methods. This combination is usually adopted to meet the requirements of a specific application.

In particular, game-theoretic approaches for multi-drone task allocation generally [26]: (i) demonstrate greater efficiency with respect to other decentralized approaches, producing suboptimal solutions that are closer to the optimal

one; (ii) have comparable or lower complexity with respect to CBBA approaches and are suitable for large-scale systems; (iii) have limited communication burden, especially in the case of competitive algorithms (i.e., using non-cooperative game settings) [30].

In regard to multi-drone route planning, several detailed surveys are available [31], [32], [33], [34], [35], mostly focusing on drone delivery systems in logistic applications. These aim at determining the joint optimal routes of teams of drones for the delivery at specific customer points, starting from a given depot or from a set of depots. Most of the works deal with the extension of classical routing problems for designing a multi-drone route planner, such as: the Traveling Salesman Problem with Drones (TSP-D), including the Flying Sidekick Traveling Salesman Problem (FSTSP), the Parallel Drone Scheduling TSP (PDSTSP) and the multiple FSTSP (mFSTSP) [36], [37], [38]; the Vehicle Routing Problem with Drones (VRP-D) [39]. Even if centralized exact solutions [36], [37], [38] and bio-inspired approaches [33] are available, the trend is to move towards a decentralized and multi-agent planner for multi-drone routing [34], [40].

From the review of multi-drone task and route planning approaches, in addition to the traditional challenges (local optima, ungranted completeness, slow convergence), new challenges have come up in connection with:

- Fault tolerance - This topic has not been sufficiently considered in classical methods [34]. Some works mention real-time implementations, but they largely focus on computational efficiency and faster convergence for online computation. Instead, they do not consider updates based on drone and environment status.
- Inaccurate target assignment - Typically, multi-drone planners execute target assignment (i.e., task allocation) first and then route planning, based on the results of target assignment. However, this strategy may lead to inaccurate target assignment, especially in dynamic and uncertain environments [41]. Indeed, if the routing aspects are ignored, target-assignment planning may be based on a too simplified view on task demands and their spatial distributions. Clearly, such inaccuracy propagates to the route-planning step.

To overcome these emerged challenges, multi-drone mission planners need to perform autonomous target assignment and route planning simultaneously and jointly in a dynamic setting. However, there are only a few works about this topic [41]. For example, some researchers have discussed the combined problem of multi-agent target assignment and route planning for the Multi-Agent Pickup and Delivery (MAPD) problem, where a team of robots has to transport a set of objects, each from an initial location and each to a specified target location. Reference [42] implements some heuristic and metaheuristic strategies to solve this problem, based on actual delivery costs. Instead, reference [41] proposes a Deep Reinforcement Learning (DRL) framework for multi-drone target assignment and path planning, in order to minimize the total flight path length under the constraints of targets' completely assignment and collision-free.

To the best of the authors' knowledge, there are no works applying game-theoretic frameworks for the multi-drone mission planning, meant as combined task and route planning, in spatio-temporal sensing with tasks having possible different priorities. A similar work to ours is proposed in [43], but the action selection to be negotiated by each drone refers to the next movement, and not to a full route plan spanning to the overall mission.

III. PROBLEM STATEMENT

This section states the reference problem by describing the application scenario and by providing the formalization of the related problem.

A. APPLICATION SCENARIO

Our reference application is a multi-drone spatio-temporal sensing mission, where: (i) ''spatio-temporal sensing'' means the execution of sensing tasks to collect reference spatiotemporal data in sites (waypoints) of interest; (ii) ''multidrone'' means the exploitation of a drone team to attain a cooperative autonomous behaviour concerning the region sensing. We assume homogeneity of the drones in the team, both for their sensing capability and for their dynamic features. The mission scenario consists in a set of sensing tasks, possibly based on a risk map, which provides the risk values of the reference event to sense within the ROI. Thus, the risk map enables a priority-based sensing of the ROI.

The drones in the team are equipped with the specific payload required for the sensing. At each moment, a drone may sense (i.e., acquire the reference sensing data of) a ground region, representing the sensor footprint, which has size depending on the flight height (i.e., the elevation with respect to ground). For the sake of convenience and in analogy with other works [44], we assume the following mission requirements:

- the ROI and each sensor footprint are respectively rectangular and square surfaces with fixed dimensions;
- all the drones move in a plane, at the same constant height and with the same constant cruise speed;
- the ROI may be discretized in square cells, named sensing cells, with the same size of the sensor footprint;
- the risk map is a spatial function that associates a priority or risk number to each sensing cell of the ROI;
- each sensing cell is associated to a sensing waypoint, which is the ground projection of the cell center with respect to the flight height of the team;
- a drone has to loiter on a sensing waypoint for a given time interval in order to successfully sense the associated cell.

The proposed assumptions are graphically depicted in Fig. 1 and Fig. 2. In detail, the concepts of risk map, sensing cells and sensing waypoints imply the definition of a sensing matrix, as illustrated in Fig. 2. Instead, Fig. 3 shows the overall system workflow for the application. In detail, a Ground

Control Station (GCS) contains all the command and control tools and the mission monitoring facilities, including a specific interface allowing the supervising of the mission operator. Such operator supplies two main inputs: (i) the definition of the ROI in terms of a risk map with respect to the reference event to be sensed; (ii) the mission template, which provides the additional user requests for the mission, e.g., cruise speed, flight height, loiter times, fixed obstacles, no-fly zones, etc. These inputs are processed by the on-Ground Mission Manager (GMM), which is a specific on-ground software module in charge of the computation of the sensing matrix and of the additional multi-drone mission settings to be delivered to the on-Board Team Mission Planner (BTMP). Note that the sensing matrix represents a full-fledged basis to then typify the mission action space as a route graph.

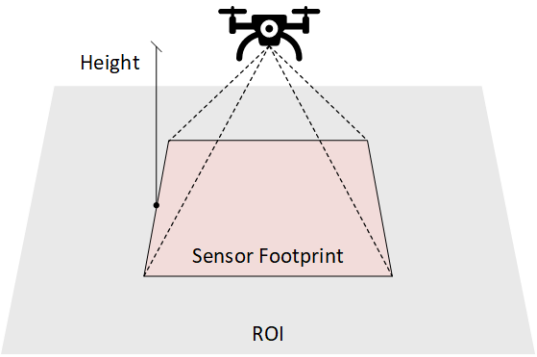


FIGURE 1. Drone and sensor footprint in a region of interest.

Fig. 4 (left) shows the high-level system architecture, which implements a decentralized solution for the team mission planning: there are no central mediators for planning, but every drone is equipped with its mission planner, which interacts with the mission planners of the other drones in a "peer-to-peer" way. Thus, the BTMP is structured as a MAS, with agents represented by the individual mission planners. The communication infrastructure includes data links between the GCS and the drones, and a Flying Ad-Hoc Network (FANET), by means of which drones in the team exchange data related to the mission planning process. Thus, the mission planners represent the FANET nodes, establishing a cooperative network for the processing of task allocations and route plans in the team. Note that the diagram in Fig. 4 (left) reports just a high-level specification of the architecture, whereas the topology of the network may implement several data routing mechanisms. For example, some specific drones in the team may play the role of backbone nodes, acting as gateways for data relaying between the GCS and other drones in the network [45].

Fig. 4 (right) illustrates the block diagram of the planning hierarchy for the single-drone view. The mission planner is decomposed in two hierarchical blocks: (i) the task planner, which is the higher-level mission planner to produce the selection and the ordering of the waypoints for the mission

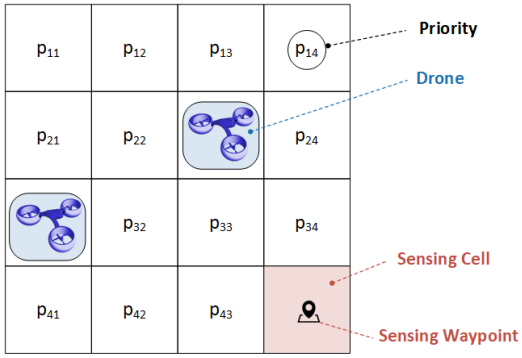


FIGURE 2. Sensing matrix for priority-based multi-drone sensing.

accomplishment; (ii) the route planner, which is the lower-level mission planner to process globally optimal routes amongst the assigned waypoints. The former acts according to the team-planning level, interacting with the distributed mission planners within the team, based also on the mission goals (i.e., sensing matrix and additional multi-drone mission settings) and on the feedbacks coming from the route planner. The latter acts according to the drone-planning level, interacting with the autopilot (i.e., the flight control software) to allow for real-time reactions to environmental changes. However, in our work, the distinction between task planner and route planner is only logical and the two related planning problems are jointly solved.

Lastly, we assume that drones are not equipped with preloaded mission plans, which shall be online generated and selected in a planning and re-planning scenario. Moreover, in the remainder of the paper, the words agents, mission planners and drones are interchangeably used.

B. PROBLEM FORMALIZATION

For a given mission, a ROI, denoted with $\mathbb{W}$, is the portion of space identified by a certain Earth surface and a variable height where the mission itself must be carried out. The mission, denoted with $\mathbb{T}$, is a set $\{\mathbf{t}_1, \cdots, \mathbf{t}_{N_t}\}$ of N_t sensing tasks, where each task $\mathbf{t}_n = \langle \mathbf{wp}_n, p_n \rangle$ is a pair composed of the related sensing waypoint $\mathbf{wp}_n$ and the priority p_n associated with it. In the following, we will interchangeably use the terms (sensing) waypoint, cell, task and target. The drone team, denoted with $\mathbb{D}$, is a set $\{\mathbf{d}_1, \cdots, \mathbf{d}_{N_d}\}$ of N_d drones, tasked with carrying out the mission $\mathbb{T}$. Even if each drone $\mathbf{d}_i$ may exhibit a cruise speed v_i, this is considered a constant v for all the drones, coherently with the homogeneity assumption for the team. Furthermore, the physics-based modelling of the drones, we adopt a point-mass model, which approximates the vehicle evolution as a single point with a mass [46].

A mission plan $\mathcal{M}_i(\mathbb{T}) = \mathcal{M}_i$ for the ith drone is defined as a sequence of sensing tasks assigned to the drone itself, i.e.,

$$\mathcal{M}_i = \left(\mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \ldots, \mathbf{t}_{i_k}\right), \quad \text{with } \mathbf{t}_{i_m} \in \mathbb{T} \quad \forall m \in \{1, 2, \ldots, k\}, k \leq N_t, \tag{1}$$

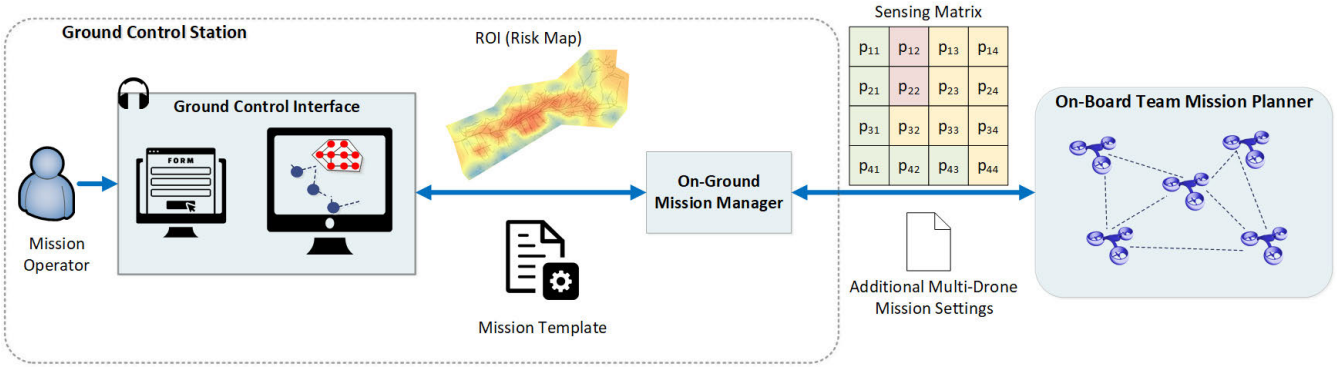


FIGURE 3. Overall system workflow for team management in multi-drone spatio-temporal sensing.

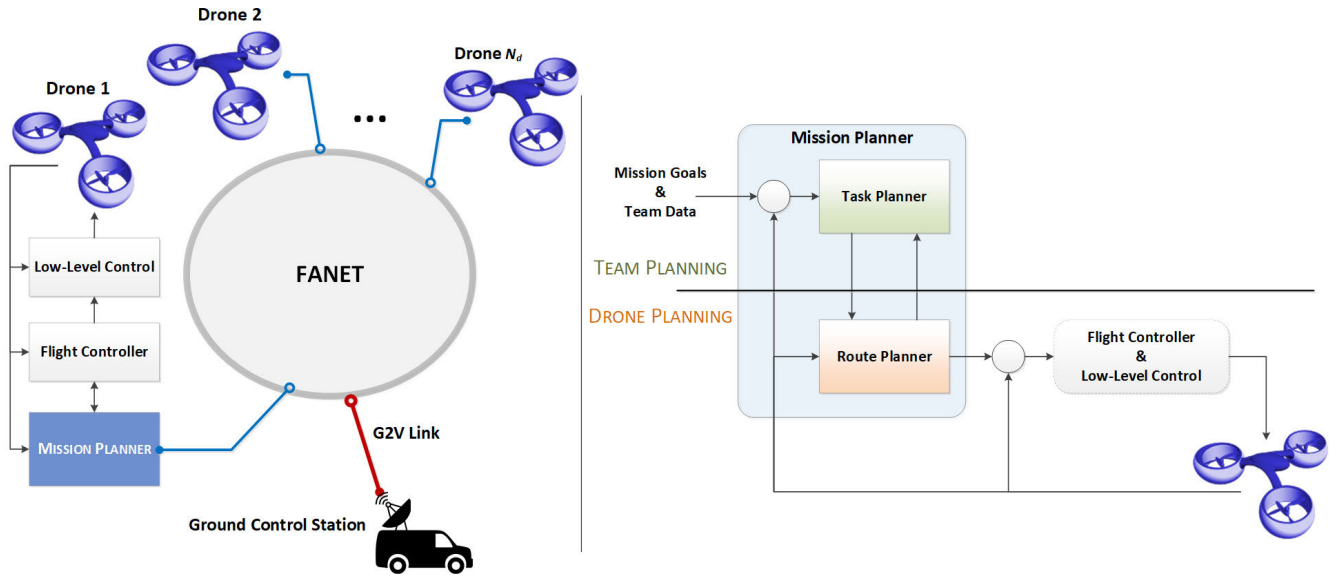


FIGURE 4. (left) High-level system architecture; (right) Block diagram of the planning hierarchy (single-drone view).

Thus, $\mathbf{t}_{i_m}$ is the mth task assigned to i in $\mathcal{M}_i$. The mission plan in (1) is an ordered task plan, that is: (i) i shall carry out $\mathbf{t}_{i_1}$, then $\mathbf{t}_{i_2}$, then $\mathbf{t}_{i_3}, \ldots$, then $\mathbf{t}_{i_k}$, if it selects $\mathcal{M}_i$; (ii) $\mathcal{M}_i$ includes only the waypoints associated to the mission tasks; (iii) the associated route of $\mathcal{M}_i$ is defined by the sequence of edges connecting consecutive waypoints in the plan and by the loitering times on each waypoint. In general, such plan includes also intermediate waypoints, which are inserted to avoid fixed obstacles and no-fly zones. Furthermore, the mission plan $\mathcal{M}_i$ is named admissible if it satisfies all the additional constraints related to the mission execution (e.g., drone endurance, mission duration, etc.). In the remainder of this work, we will denote with $\mathcal{M}_i$ an admissible mission plan for the ith drone.

The mission plan $\mathcal{M}_i$ can also be interpreted as a possible action that the ith agent may deliberate, where the action is the route assigned to the drone for mission execution. The set of all admissible actions of i is the set $\mathbb{M}_i(\mathbb{T}) = \mathbb{M}_i$, composed of all admissible mission plans that the ith drone may carry out during the mission. Since the sensing tasks are finite, the cardinality of $\mathbb{M}_i$ is finite and limited by the following relation,

$$|\mathbb{M}_i| \leq \sum_{k=1}^{N_t} \frac{N_t}{(N_t - k)}, \forall i \in \{1, \ldots, N_d\}. \tag{2}$$

A joint mission plan $\mathcal{M}_{\mathbb{D}}(\mathbb{T}) = \mathcal{M}$ is defined as the N_d-set of the plans assigned to each drone of the team $\mathbb{D}$, i.e.,

$$\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_{N_d}\}, \tag{3}$$

and it belongs to the joint set of the drone action spaces $\mathbb{M} = \mathbb{M}_1 \times \mathbb{M}_2 \times \ldots \times \mathbb{M}_{N_d}$.

In order to evaluate the proposed solution, a sensing performance index is introduced resorting to the concept of Expected Cost of Ignorance (ECI). In detail, if E is the reference event to be sensed, the not visitation cost of a waypoint $\mathbf{wp}_n$ within a given time interval is related to the occurrence probability $P_{E,n}$ and the occurrence impact $C_{E,n}$ of E over $\mathbf{wp}_n$ in that time interval. Probability and impact functions

depend on targets in the risk map, i.e., they allow a ranking of their priority based on their risk figures. The ECI of a task $\mathbf{t}_n$ according to a joint mission plan $\mathcal{M}$ is given by,

$$\mathrm{ECI}_{\mathcal{M}}(\mathbf{t}_n) = \int_0^{t_{\mathcal{M},n}} P_{\mathrm{E},n}(t) C_{\mathrm{E},n}(t) dt, \tag{4}$$

where $t_{\mathcal{M},n}$ is the first instant of time in which the waypoint of $\mathbf{t}_n$ is visited according to $\mathcal{M}$. Note that the definition of $t_{\mathcal{M},n}$ as the first visitation instant implies that, in case of overlapping on $\mathbf{t}_n$ in $\mathcal{M}$ (i.e., the task $\mathbf{t}_n$ is assigned to more than one drone in $\mathcal{M}$), only the assignment corresponding to $t_{\mathcal{M},n}$ will affect the ECI value of $\mathbf{t}_n$. Thus, the coverage of a task with more drones will not provide an improvement (i.e., a reduction) of the ECI. Note that, according to the definition of the ECI operator in (4), the routing space of each drone may be represented as a weighted graph, whose edges connect waypoints and have weights equal to the variation of $\mathrm{ECI}_{\mathcal{M}}$ in case that edge is crossed for a mission plan.

The ECI of the mission $\mathbb{T}$ related to a joint mission plan $\mathcal{M}$ is given by,

$$\mathrm{ECI}_{\mathcal{M}}(\mathbb{T}, \mathbb{D}) = \sum\nolimits_{\mathbf{t}_n \in \mathbb{T}} \mathrm{ECI}_{\mathcal{M}}(\mathbf{t}_n), \tag{5}$$

In order to minimize the ECI of the mission, an optimal joint mission plan $\mathcal{M}^{\mathrm{opt}}$ must be considered, therefore, the following optimization problem must be solved,

$$\mathcal{M}^{\mathrm{opt}} = \underset{\mathcal{M} \in \mathbb{M}}{\mathrm{argmin}} \mathrm{ECI}_{\mathcal{M}}(\mathbb{T}, \mathbb{D}) \tag{6}$$

The problem in (6) represents a constrained stochastic optimization problem and extends some well-known problems, such as the vehicle-target assignment, the multivehicle motion planning and the multi-agent task allocation.

IV. GAME-THEORETIC SOLUTION

This section describes the game-theoretic solution of the problem in (6). Such solution is structured in three steps: (i) the design of the game to be played by the agents; (ii) the design of the learning-in-games method; (iii) the implementation of the coordination algorithm.

A. GAME DESIGN

To solve the problem in (6) by means of a game-theoretic approach, the first step is to design the game model to define the interaction structure amongst the players (i.e., the agents). Within this work, we adopt a competitive paradigm underlying the game model: the agents are antagonistic (selfish or self-interested), namely, they have distinct individual objectives (that are potentially in conflict with each other) and independently act according to their local information [47]. Competitive agents fit better with scalable and resilient contexts: in spite of its internal competitive nature, a MAS with selfish agents may effectively control a distributed system with a single global objective and may provide the degree of robustness and flexibility that are needed in classic large distributed applications [48]. In our case, an artificial competition is set amongst the selfish agents representing mission planners, which cannot be real opponents because their interaction shall achieve a global goal, namely, the minimization of the ECI. Based on this competitive paradigm, a noncooperative game is designed. Indeed, noncooperative game theory is the study of games wherein the players independently make decisions and is well suited for competitive MAS, providing a tool to explicitly model both the internal processes of individual agents and their interactions with others (i.e., how their choices mutually affect others) [48].

Thus, utility functions are introduced to model both individual behaviours and global interactions of the agents in a competitive utility-oriented setting. We start designing a global (i.e., agent-independent) function $G = G(\mathcal{M})$ to be maximized, which captures the desired behaviour for the overall team $\mathbb{D}$ to accomplish $\mathbb{T}$ and represents the interaction structure amongst the mission planners. This function is also named mission utility and is defined as,

$$G(\mathcal{M}) = -\mathrm{ECI}_{\mathcal{M}}(\mathbb{T}, \mathbb{D}) = -\sum\nolimits_{\mathbf{t}_n \in \mathbb{T}} \mathrm{ECI}_{\mathcal{M}}(\mathbf{t}_n). \tag{7}$$

Instead, individual utility functions capture the preferences of the agents over their individual mission plans and are designed exploiting the concept of wonderful life utility or marginal contribution utility [49]. In detail, the following function u_i represents the individual utility of the ith drone,

$$u_i(\mathcal{M}) = G(\mathcal{M}) - G\left(\mathcal{M}_i^{\emptyset}\right), \tag{8}$$

where $\mathcal{M}_i^{\emptyset}$ is the joint mission plan $\mathcal{M}$ without considering any action for the ith drone, i.e., with $\mathcal{M}_i$ as the empty sequence. With a such utility function, the proposed game model is an exact potential game [50] with $G(\mathcal{M})$ as potential function, as proven in [51] for the games following the utility structure based on wonderful life utility. This means that, when a player i changes its plan, the variation in its utility function u_i is equal to the variation in the global utility G, i.e., the individual utility improvement of each player is equal to the improvement in the global utility.

Potential games are a particular class of non-cooperative games provided with [52]: (i) at least a Nash equilibrium, which coincides with the optima (even local) of the potential function; (ii) the finite improvement property, which implies that any sequence of improving players' moves converges to a Nash equilibrium in finite time. Thus, any coordination algorithm that converges to a Nash equilibrium of the potential game, will also converge to the optima of $G(\mathcal{M})$ and, equivalently, of $\mathrm{ECI}_{\mathcal{M}}(\mathbb{T})$, solving the problem in (6).

B. LEARNING DESIGN

The next step aims at designing a method to solve the game model (i.e., to find a Nash equilibrium) in a dynamic and decentralized setting. The proposed method relies on a learning-in-games approach. The topic of learning in games addresses the issue of dynamic processes leading to equilibrium by means of learning schemes, where players learn about the environment (including the behaviour of other

players). In the setup of learning in games [53]: (i) players repetitively play a game over a sequence of stages; (ii) at each stage, players apply a learning rule, using past experiences/observations to select a strategy for the current stage; (iii) once player strategies are selected, the game is played, information is updated, and the process is repeated. This process may effectively suggest an online algorithm to dynamically reach a Nash equilibrium, exploiting specific learning rules [54].

In our case, considering the planning and re-planning scenario highlighted in section III.A, a learning rule must be designed for embodying the notion of equilibria selection in potential games, i.e., to guarantee convergence to the most efficient Nash equilibrium. A learning rule that could serve this purpose is the Log-Linear Learning (LLL) [55]. By introducing random perturbations into the decision-making process, such rule allows the selection of suboptimal actions to avoid the convergence to local minima of G. In potential games, LLL guarantees that only the joint action profiles that maximize the potential function are stochastically stable [55]. The structure of its learning strategy has the following settings:

- asynchrony - players update their actions one at a time;
- decomposition - the learning rule is independent of the utility function;
- decoupling - the actions of an agent depend on its individual utility function, but not on the utility functions of other agents (only the past stream of previous actions of itself and of other players is used);
- completeness - at any stage, a player can select any action in the related action set.

Considering the application under exam, the first three settings may be easily implemented. On the contrary, completeness entails a tough implementation challenge because of the size of the sets $\mathbb{M}_i$, which increase with N_t according to (2). Indeed, for each drone, LLL requires the evaluation of the utility function for each mission plan in $\mathbb{M}_i$ and, consequently, the design of a practical online implementation is computationally intractable also for small values of N_t.

A way to tackle the completeness issue is to exploit the Binary LLL (BLLL) [55], i.e., a variant of LLL which allows to consider a constrained action set $\mathbb{C}_i$ (rather than the entire $\mathbb{M}_i$) for the ith agent, maintaining the prerogative that only potential function maximizers are stochastically stable. More in general, $\mathbb{C}_i$ is a subset of the action space $\mathbb{A}_i$ of the ith agent ($\mathbb{C}_i \subseteq \mathbb{A}_i$), which depends on the action selected at the preceding step by i. By convention, $a_i \in \mathbb{C}_i(a_i)$ for any action $a_i \in \mathbb{A}_i$ [55], that is a player is allowed to stay with its previous action. Furthermore, $\mathbb{C}_i$ must satisfy the following two properties [56]:

- reachability - for each agent i and any pair of action $a_i^0, a_i^m \in \mathbb{A}_i$, there exists a sequence of actions $a_i^0 \to a_i^1 \to \cdots \to a_i^{m-1} \to a_i^m$ satisfying $a_i^k \in \mathbb{C}_i\left(a_i^{k-1}\right) \forall k \in \{1, \ldots, m\}$;
- reversibility - for each agent i and any pair of action $a_i^0, a_i^1 \in \mathbb{A}_i$, if $a_i^1 \in \mathbb{C}_i\left(a_i^0\right)$, then $a_i^0 \in \mathbb{C}_i\left(a_i^1\right)$.

The BLLL learning structure is the following [55]:

1. at each time step t, a player i is randomly chosen and is allowed to alter its action while all other agents repeat their previous actions;
2. the player i uniformly (or with a given distribution probability) chooses a new possible action a_i^* within $\mathbb{C}_i(a_i(t-1))$, where $a_i(t-1)$ is the previous action selected by i;
3. the player i selects a_i^* as its new action $a_i(t)$ according to the following probabilities,

$$P(a_i(t) = a_i(t-1)) = \frac{e^{\frac{1}{\tau}u_i(a(t-1))}}{e^{\frac{1}{\tau}u_i(a(t-1))} + e^{\frac{1}{\tau}u_i(a_i^*, a_{-i}(t-1))}}, \quad (9)$$

$$P\left(a_i(t) = a_i^*\right) = \frac{e^{\frac{1}{\tau}u_i(a_i^*, a_{-i}(t-1))}}{e^{\frac{1}{\tau}u_i(a(t-1))} + e^{\frac{1}{\tau}u_i(a_i^*, a_{-i}(t-1))}}, \quad (10)$$

where $a(t-1)$ is the joint action set at the step $(t-1)$, $a_{-i}(t-1)$ is the joint action set played by all the players except for i, and τ (with $\tau > 0$) is a parameter called temperature.

In order to apply the BLLL to our application, we need to define a filtering rule for the ith agent to determine its constrained action set at the step t, based on the given mission plan deliberated by i at $t-1$. For this purpose, we introduce some definitions reported hereinafter.

Given a mission plan $\mathcal{M}_i \in \mathbb{M}_i$ as a sequence of k tasks, another plan $\mathcal{M}_i^{[+1]}$ is an immediate descendant of $\mathcal{M}_i$ if $\mathcal{M}_i^{[+1]} = \mathcal{M}_i + \mathbf{t}_{i_{k+1}} \triangleq \left(\mathbf{t}_{i_1}, \ldots, \mathbf{t}_{i_k}, \mathbf{t}_{i_{k+1}}\right)$, with $\mathbf{t}_{i_{k+1}} \in \mathbb{T} - \mathbb{T}_i$, where $\mathbb{T}_i = \left\{\mathbf{t}_{i_1}, \ldots, \mathbf{t}_{i_k}\right\}$, and with $\mathcal{M}_i^{[+1]}$ being an admissible plan. Likewise, a plan $\mathcal{M}_i^{[-1]}$ is the immediate ascendant (or ancestor) of $\mathcal{M}_i$ if $\mathcal{M}_i^{[-1]} = \mathcal{M}_i - \mathbf{t}_{i_k} \triangleq \left(\mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \ldots, \mathbf{t}_{i_{k-1}}\right)$ with $k \geq 1$, where $\mathbf{t}_{i_k}$ is the last task of the sequence in $\mathcal{M}_i$.$\mathcal{M}_i$ is also named parent of $\mathcal{M}_i^{[+1]}$, and $\mathcal{M}_i^{[-1]}$ is parent of $\mathcal{M}_i$. More generally, $\mathcal{M}_i^{[+m]}$ is an mth order descendant of $\mathcal{M}_i$ if,

$$\begin{aligned}\mathcal{M}_i^{[+m]} &= \mathcal{M}_i + \mathbf{t}_{i_{k+1}} + \cdots + \mathbf{t}_{i_{k+m}} = \\ &= \left(\mathbf{t}_{i_1}, \ldots, \mathbf{t}_{i_k}, \mathbf{t}_{i_{k+1}}, \ldots, \mathbf{t}_{i_{k+m}}\right),\end{aligned}$$

with:

$$\begin{aligned}&\mathbf{t}_{i_{k+1}} \in \mathbb{T} - \mathbb{T}_i \\ &\mathbf{t}_{i_{k+2}} \in \mathbb{T} - \mathbb{T}_i - \left\{\mathbf{t}_{i_{k+1}}\right\}, \ldots \\ &\mathbf{t}_{i_{k+m}} \in \mathbb{T} - \mathbb{T}_i - \left\{\mathbf{t}_{i_{k+1}}, \ldots, \mathbf{t}_{i_{k+m-1}}\right\} \\ &\mathcal{M}_i^{[+m]} \text{ is an admissible plan.}\end{aligned} \quad (11)$$

$\mathcal{M}_i^{[-m]}$ is the mth order ascendant (or ancestor) of $\mathcal{M}_i$ if,

$$\begin{aligned}\mathcal{M}_i^{[-m]} &= \mathcal{M}_i - \mathbf{t}_{i_{k-(m-1)}} - \mathbf{t}_{i_{k-(m-2)}} - \cdots - \mathbf{t}_{i_k} = \\ &= \left(\mathbf{t}_{i_1}, \mathbf{t}_{i_2}, \ldots, \mathbf{t}_{i_{k-m}}\right), \quad k \geq m\end{aligned} \quad (12)$$

If $m \geq k$, we assume $\mathcal{M}_i^{[-m]}$ to be the empty sequence. If $\mathcal{M}_i$ is the empty sequence, then $\mathcal{M}_i$ has no ascendants.

Consequently, a mission plan $\mathcal{M}_i \in \mathbb{M}_i$ with k tasks has k ancestors (including the empty sequence) and at most $\sum_{n=1}^{\bar{k}} \frac{\bar{k}}{(\bar{k}-n)}$ descendants, where $\bar{k} = |\mathbb{T}| - k$. Finally, given the ith agent, we respectively denote with $\mathbb{F}_i^{(m)}(\mathcal{M}_i)$ and $\mathbb{B}_i^{(m)}(\mathcal{M}_i)$ the set of descendant (or forward) plans and the set of ascendant (or backward) plans of $\mathcal{M}_i$ with order until m. Thus,

$$\mathbb{F}_i^{(m)}(\mathcal{M}_i) = \left\{ \mathbb{M}_i^{[+j]}(\mathcal{M}_i) \mid j = 1, \cdots, m \right\}, \quad (13)$$

$$\mathbb{B}_i^{(m)}(\mathcal{M}_i) = \left\{ \mathcal{M}_i^{[-1]}, \mathcal{M}_i^{[-2]}, \ldots, \mathcal{M}_i^{[-m]} \right\}, \quad (14)$$

where $\mathbb{M}_i^{[+j]}(\mathcal{M}_i)$ is the set of all the descendant plans of order j for the mission plan $\mathcal{M}_i$.

Let $\mathcal{M}_i(t)$ be the mission plan (i.e., the action) selected by the agent i at the time step t. For the purposes of our learning method, we design the constrained action set of $\mathcal{M}_i(t)$ as,

$$\mathbb{C}_i^w(\mathcal{M}_i(t)) = \mathbb{B}_i^{(w)}(\mathcal{M}_i(t)) \cup \mathcal{M}_i(t) \cup \mathbb{F}_i^{(w)}(\mathcal{M}_i(t)), \quad (15)$$

where $w \leq |\mathbb{T}|$ is a parameter named planning window, indicating the maximum order of descendants and ascendants of $\mathcal{M}_i(t)$ that are part of its constrained action set. In other words, at each time step, w represents the maximum number of tasks which could be added or subtracted to or from $\mathcal{M}_i(t)$ to generate the set $\mathbb{C}_i^w(\mathcal{M}_i(t))$ of possible new mission plans for i.

Then, the agent i may randomly (e.g., uniformly) select a mission plan in $\mathbb{C}_i^w(\mathcal{M}_i(t))$ as its updated action for the time step $t+1$. Furthermore, if $k(t)$ is the number of tasks of $\mathcal{M}_i(t)$ and $\bar{k}(t) = |\mathbb{T}| - k(t)$, at most $\bar{k}(t)$ tasks can be added to $\mathcal{M}_i(t)$ when $w > \bar{k}(t)$, and at most $k(t)$ tasks can be subtracted from $\mathcal{M}_i(t)$ when $w > k(t)$. Fig. 5 shows an application example of the designed filtering rule in (15) for the learning process.

The filtering rule in (15) complies with the required properties of a constrained action set for BLLL, considering the association between actions a_i and plans $\mathcal{M}_i$. Firstly, the set $\mathbb{C}_i^w(\mathcal{M}_i)$ satisfies the relation $\mathcal{M}_i \in \mathbb{C}_i^w(\mathcal{M}_i)$ as per construction of $\mathbb{C}_i^w(\mathcal{M}_i)$. Moreover, it complies with the reachability and reversibility properties. The redefinition of the above properties in the case of drones' mission plans are shown by the following theorems:

- Theorem 1 - Reachability: for any drone i and any pair of mission plans $\mathcal{M}_i^0, \mathcal{M}_i^m \in \mathbb{M}_i$, there exists a sequence of mission plans $\mathcal{M}_i^0 \to \mathcal{M}_i^1 \to \cdots \to \mathcal{M}_i^{m-1} \to \mathcal{M}_i^m$ satisfying $\mathcal{M}_i^k \in \mathbb{C}_i^w\left(\mathcal{M}_i^{k-1}\right) \quad \forall k \in (1, \ldots, m)$.
- Theorem 2 - Reversibility: for any drone i and any pair of mission plans $\mathcal{M}_i^0, \mathcal{M}_i^1 \in \mathbb{M}_i$, if $\mathcal{M}_i^0 \in \mathbb{C}_i^w\left(\mathcal{M}_i^1\right)$, then $\mathcal{M}_i^1 \in \mathbb{C}_i^w\left(\mathcal{M}_i^0\right)$.

The proofs of these theorems are respectively reported in Appendix A and Appendix B. Thus, our proposed filtering rule in (15) represents a proper basis for an ad hoc learning method, relying on BLLL, to be adopted by the proposed mission planners.

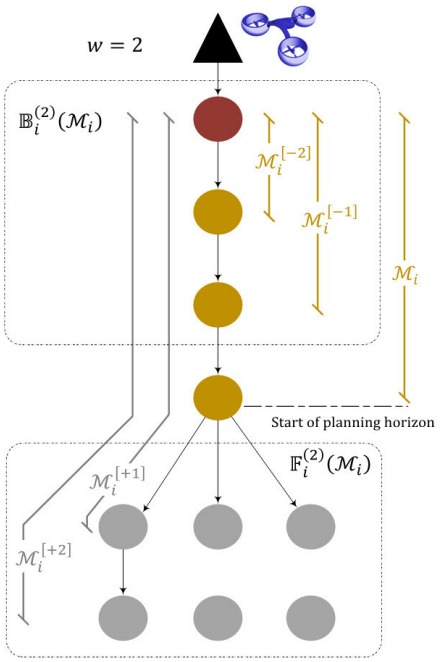


FIGURE 5. Application example of the designed filtering rule in the learning process.

C. COORDINATION ALGORITHM

Based on the filtering rule in (15), the final learning rule for our coordination algorithm requires the determination of the distribution probability to select the new mission plan $\mathcal{M}_i^* \in \mathbb{C}_i(\mathcal{M}_i(t-1))$ at the time step t. In our case, a uniform distribution (that is the distribution considered in the basic version of BLLL) cannot be efficiently applied considering how $\mathbb{C}_i^w(\mathcal{M}_i(t-1))$ is built. Indeed, the number of descendant mission plans is generally much larger than the number of ancestor mission plans. Therefore, if a uniform distribution is used for selecting $\mathcal{M}_i^*$, the sequence of joint mission plans will likely include random sequences of sensing tasks with many overlaps (i.e., tasks assigned to more than one drone), which do not represent a benefit for $G(\mathcal{M})$ as per definition of the ECI operator in (4).

To overcome this issue, we propose the adoption of a random distribution weighted on the agents' individual utility functions. To define a suitable weighing for a such distribution, we need to investigate how the evolution of the learning in the game is influenced by the global function $G(\mathcal{M})$ in (7).

For this purpose, we suppose that $P_{\mathrm{E},n}(t)$ and $C_{\mathrm{E},n}(t)$ are constant and equal to 1 for each waypoint $\mathbf{wp}_n$. Anyway, these assumptions are introduced just for the sake of illustration, and the proposed approach is the same in the case of time- and waypoint-variable functions of $P_{\mathrm{E},n}(t)$ and $C_{\mathrm{E},n}(t)$. If the mission duration is equal to $\mathcal{T}$, the global function $G(\mathcal{M})$ is given by,

$$G(\mathcal{M}) = -\sum_{\mathbf{t}_n \in \mathbb{T}} \mathrm{ECI}_{\mathcal{M}}(\mathbf{t}_n) = -\sum_{\mathbf{t}_n \in \mathbb{T}} t_{\mathcal{M},\mathbf{t}_n}, \quad (16)$$

where $t_{\mathcal{M},\mathbf{t}_n}$ is the assignment time of the sensing task $\mathbf{t}_n$, i.e., the time when $\mathbf{t}_n$ is carried out according to $\mathcal{M}$. If a task

$\mathbf{t}_u$ is not assigned in $\mathcal{M}$, the related assignment time $t_{\mathcal{M},\mathbf{t}_u}$ will be equal to $\mathcal{T}$.

Given the joint plan $\mathcal{M}$(t), the individual plan $\mathcal{M}_i$(t) of the drone i, and the set of individual plans $\mathcal{M}_{-i}$(t) of the other drones in the team $\mathbb{D}$, we denote with:

- N_a the number of tasks that are included in $\mathcal{M}_i$(t) and are not included in $\mathcal{M}_{-i}$(t), identified by the set $\mathbb{T}_\mathrm{a} = \{\mathbf{t}_{\mathrm{a}_1}, \cdots, \mathbf{t}_{\mathrm{a}_{N_\mathrm{a}}}\}$;
- N_c the number of tasks that are included both in $\mathcal{M}_i$(t) and in $\mathcal{M}_{-i}$(t), and that are carried out with a greater utility in $\mathcal{M}_i$(t) than in $\mathcal{M}_{-i}$(t), identified by the set $\mathbb{T}_\mathrm{c} = \{\mathbf{t}_{\mathrm{c}_1}, \cdots, \mathbf{t}_{\mathrm{c}_{N_\mathrm{c}}}\}$;
- N_r the number of tasks that are (i) not included in $\mathcal{M}_i$(t) and are included in $\mathcal{M}_{-i}$(t), or (ii) that are included both in $\mathcal{M}_i$(t) and in $\mathcal{M}_{-i}$(t) and are carried out with a greater utility in $\mathcal{M}_{-i}$(t), identified by the set $\mathbb{T}_\mathrm{r} = \{\mathbf{t}_{\mathrm{r}_1}, \cdots, \mathbf{t}_{\mathrm{r}_{N_r}}\}$;
- N_u the number of tasks that are not included neither in $\mathcal{M}_i$(t) nor in $\mathcal{M}_{-i}$(t), identified by the set $\mathbb{T}_\mathrm{u} = \{\mathbf{t}_{\mathrm{u}_1}, \cdots, \mathbf{t}_{\mathrm{u}_{N_\mathrm{u}}}\}$.

In other words, at the time step t : N_a is the number of tasks that are allocated only to i; N_c is the number of tasks that are allocated to i and also to other drones, but that are carried out with a greater utility by i (or that i carries out earlier, in the case of the assumptions introduced for $P_{\mathrm{E},n}$ and $C_{\mathrm{E},n}$); N_r is the number of tasks that are allocated to some drones that are different from i, or that are allocated to i and also to other drones, but that are carried out with a greater utility by other drones; N_u is the number of tasks that are not allocated at all. With these notations, the individual utility function of the ith drone can be written as,

$$u_i(\mathcal{M}(t)) = G(\mathcal{M}(t)) - G\left(\mathcal{M}_i^{\emptyset}(t)\right) = \\ - \sum\nolimits_{\mathbf{t}_j \in \mathbb{T}_a} t_{\mathcal{M}_i(t),\mathbf{t}_j} + N_a\mathcal{T} - \sum\nolimits_{\mathbf{t}_j \in \mathbb{T}_c} t_{\mathcal{M}_i(t),\mathbf{t}_j} + \\ + \sum\nolimits_{\mathbf{t}_j \in \mathbb{T}_c} t_{\mathcal{M}_i^{\Phi}(t),\mathbf{t}_j}. \quad (17)$$

We suppose that the ith drone is allowed to alter its mission plan at the step $t+1$ according to the BLLL rules, choosing a new plan in the constrained set $\mathbb{C}_i^w(\mathcal{M}_i(t))$. Just for the sake of illustration, we also assume $w = 1$. We can demonstrate that the selectable mission plans $\mathcal{M}_i(t+1) \in \mathbb{C}_i^1(\mathcal{M}_i(t))$ increasing the utility of i, i.e., with $u_i(\mathcal{M}_i(t+1), \mathcal{M}_{-i}(t)) > u_i(\mathcal{M}_i(t), \mathcal{M}_{-i}(t))$ and with $\mathcal{M}_{-i}(t) = \mathcal{M}_{-i}(t+1)$, are probably only those adding a task $\mathbf{t}_\mathrm{a} \in \mathbb{T}_\mathrm{a}$ to $\mathcal{M}_i(t)$, i.e., a task not allocated to other drones in $\mathcal{M}_{-i}$(t). Indeed, the following cases may occur for the selection of $\mathcal{M}_i(t+1)$ starting from $\mathcal{M}_i(t)$ and its impact on $\Delta u_i(t+1) = u_i(\mathcal{M}_i(t+1), \mathcal{M}_{-i}(t)) - u_i(\mathcal{M}_i(t), \mathcal{M}_{-i}(t))$:

- $\mathcal{M}_i(t+1)$ is a descendant of $\mathcal{M}_i(t)$ adding a task $\mathbf{t}_a \in \mathbb{T}_\mathrm{a}$ - In this case, there is an increase of the individual utility of i equal to,

$$\Delta u_i(t+1) = \mathcal{T} - t_{\mathcal{M}_i(t+1),\mathbf{t}_a}. \quad (18)$$

- $\mathcal{M}_i(t+1)$ is a descendant of $\mathcal{M}_i(t)$ adding a task $\mathbf{t}_c \in \mathbb{T}_\mathrm{c}$ - In this case, there is an increase of the individual utility of i equal to,

$$\Delta u_i(t+1) = t_{\mathcal{M}_i^{\emptyset}(t),\mathbf{t}_c} - t_{\mathcal{M}_i(t+1),\mathbf{t}_c}. \quad (19)$$

Therefore, the task $\mathbf{t}_c$ will provide a greater increase of the individual utility of i with respect to the task $\mathbf{t}_a$ only if:

$$\left(t_{\mathcal{M}_i^{\emptyset}(t),\mathbf{t}_c} - t_{\mathcal{M}_i(t+1),\mathbf{t}_c}\right) > \left(\mathcal{T} - t_{\mathcal{M}_i(t+1),\mathbf{t}_a}\right) \Rightarrow \\ \Rightarrow t_{\mathcal{M}_i(t+1),\mathbf{t}_c} < \left(t_{\mathcal{M}_i(t+1),\mathbf{t}_a} + \varphi\right) \quad (20)$$

where $\varphi = t_{\mathcal{M}_i^{\varnothing}(t),\mathbf{t}_c} - \mathcal{T}$ is a negative quantity as per definition of $\mathcal{T}$. It is clear that the relationship in (20) is unlikely to be satisfied because the mission duration $\mathcal{T}$ is generally much larger than $t_{\mathcal{M}_i(t+1),\mathbf{t}_a}$.

- $\mathcal{M}_i(t+1)$ is a descendant of $\mathcal{M}_i(t)$ adding a task $\mathbf{t}_r \in \mathbb{T}_r$ - In this case, there is no variation of the individual utility of i, i.e., $\Delta u_i(t+1) = 0$.
- $\mathcal{M}_i(t+1) = \mathcal{M}_i(t)$ - In this case, no tasks are added and the previous plan is still selected, thus, $\Delta u_i(t+1) = 0$.
- $\mathcal{M}_i(t+1)$ is an ancestor of $\mathcal{M}_i(t)$ - In this case, it is evident that there is a possible reduction of the individual utility of i, i.e., $\Delta u_i(t+1) \leq 0$.

Therefore, if we adopt a utility-weighted distribution for the learning rule, at each time step the agent i (if allowed to alter its plan) will:

- add a task $\mathbf{t}_a \in \mathbb{T}_\mathrm{a}$ with high probability, proportionally to the term in (18);
- add a task $\mathbf{t}_c \in \mathbb{T}_c$, with lower probabilities with respect to the previous option, proportionally to the term in (19);
- add other tasks, or will select the previous plan, or will select an ancestor plan, with even lower probabilities.

Such rule is expected to provide better results with respect to the uniform distribution by drastically reducing the overlapping in agent planning, while keeping stochastic perturbations. However, it could be not yet satisfactory for a practical implementation since the tasks $\mathbf{t}_\mathrm{a} \in \mathbb{T}_\mathrm{a}$ could provide numerical values of $\Delta u_i(t+1)$ with small differences. In this case, little stochastic perturbations may significantly affect the selection of $\mathbf{t}_\mathrm{a}$, i.e., the ith drone would add a new task $\mathbf{t}_\mathrm{a}$ in a "too random" way, not rewarding enough the tasks that would increase the agent's utility and possibly cumulating this stochastic penalty in the next stages of selection.

In order to have a more balanced weighted distribution for the performance maximization, we propose a first solution of the coordination algorithm, which normalizes and then overweighs the distribution of drones' utility (over $\mathbb{M}_\mathrm{i}$) using an exponential function and a temperature β in a similar way as reported in (9) and in (10). In other words, the exponential function and the temperature allow the agent to select (if required) the new mission plans by further rewarding those with higher utilities in the constrained action set. The proposed balancing aims at achieving a satisfactory equilibrium for the game. The pseudocode of this first solution is shown in Fig. 6.

Note that the normalization of the values of the drones' utility is necessary to prevent any numerical overflows when

1: **function** drone_mission_planning$\left(i, \mathcal{M}_i^l, \mathbb{T}, \mathbb{D}, w, \alpha, \gamma, \beta, \tau, t\right)$

Input: identifier i of the drone; last mission plan proposal $\mathcal{M}_i^l$ of i (notice that $\mathcal{M}_i^l$=$\mathcal{M}_i(t-1)$); target set $\mathbb{T}$; drone team $\mathbb{D}$; planning window w; α and γ percentiles of the utilities related to the plans belonging to the constrained set (with $\alpha > \gamma$); temperatures β and τ; current time step t

Output: new mission plan proposal $\mathcal{M}_i(t)$ of i

2: $\mathcal{M}_{-i}(t-1) \leftarrow$ drone_receive_message$(\mathbb{D})$ ▷ collect proposals (if any) from a drone

3: **if** $\mathcal{M}_{-i}(t-1) = \emptyset$ **then** ▷ t is not the turn of i for proposing a new mission pla

4: $\mathbb{F}_i^{(w)}(t) =$ update_descendant_plan_library$(\mathcal{M}_i^l, w)$ ▷ resume generation of plans in the library $\mathbb{F}_i^{(w)}$ at t

5: **return** $\mathbb{M}_i(t)$

6: **else** ▷ t is the turn of i for proposing a new mission plan

7: $\mathbb{F}_i^{(w)}(t) =$ update_descendant_plan_library$(\mathcal{M}_i^l, w)$ ▷ complete the descendant plan library $\mathbb{F}_i^{(w)}$ at t

8: $\mathbb{B}_i^{(w)}(t) =$ update_descendant_plan_library$(\mathcal{M}_i^l, w)$ ▷ complete the ascendant plan library $\mathbb{B}_i^{(w)}$ at t

9: $\mathbb{C}_i^{(w)}(t) = \mathbb{B}_i^{(w)}(t) \cup \mathcal{M}_i^l \cup \mathbb{F}_i^{(w)}(t)$ ▷ compute the constrained library for new proposals

10: $\mathbb{U}_{\mathbb{C}_i^{(w)}}(t) = \left\{u_i\left(\mathcal{M}_i^j, \mathcal{M}_{-i}(t-1)\right), \mathcal{M}_i^j \in \mathbb{C}_i^{(w)}(t)\right\}$ ▷ compute the utilities considering each mission plan in the constrained library

11: $\mathbb{U}^{\mathrm{n}}_{\mathbb{C}_i^{(w)}}(t) = \left\{\dfrac{u_i - \text{percentile}(\mathbb{U}_{\mathbb{C}_i^{(w)}}(t), (\alpha+\gamma)/2)}{\text{percentile}\left(\mathbb{U}_{\mathbb{C}_i^{(w)}}(t), \alpha\right) - \text{percentile}\left(\mathbb{U}_{\mathbb{C}_i^{(w)}}(t), \gamma\right)}, u_i \in \mathbb{U}_{\mathbb{C}_i^{(w)}}(t)\right\}$ ▷ normalize the utilities using percentiles of $\mathbb{U}_{\mathbb{C}_i^{(w)}}$

12: $P_{\mathrm{c}} = \left\{p\left(\mathcal{M}_i^j, t\right) = \dfrac{e^{\frac{1}{\beta}u_i^{\mathrm{n}}\left(\mathcal{M}_i^j, \mathcal{M}_{-i}(t-1)\right)}}{\sum_{\mathcal{M}_i^k \in \mathbb{C}_i^{(w)}} e^{\frac{1}{\beta}u_i^{\mathrm{n}}\left(\mathcal{M}_i^k, \mathcal{M}_{-i}(t-1)\right)}}, \mathcal{M}_i^j \in \mathbb{C}_i^{(w)}\right\}$ ▷ compute the probability distribution of the mission plans belonging to the constrained set

13: $\mathcal{M}_i^* =$ random_selection$\left(\mathbb{C}_i^{(w)}(t), P_{\mathrm{c}}\right)$ ▷ 1st random selection according to the second step of the BLLL

14: $u_i^{max} = \max\left(u_i\left(\mathcal{M}_i^l, \mathcal{M}_{-i}(t-1)\right), u_i\left(\mathcal{M}_i^*, \mathcal{M}_{-i}(t-1)\right)\right)$ ▷ compute the maximum between the utilities related to the plans $\mathcal{M}_i^*$ and $\mathcal{M}_i^l$

15: $P_{\text{final}} = \left\{p\left(\mathcal{M}_i^j, t\right) = \dfrac{e^{\frac{1}{\tau u_i^{max}}u_i\left(\mathcal{M}_i^j, \mathcal{M}_{-i}(t-1)\right)}}{\sum_{\mathcal{M}_i^k \in \{\mathcal{M}_i^* \cup \mathcal{M}_i\}} e^{\frac{1}{\tau u_i^{max}}u_i\left(\mathcal{M}_i^k, \mathcal{M}_{-i}(t-1)\right)}}, \mathcal{M}_i^j \in \{\mathcal{M}_i^* \cup \mathcal{M}_i^l\}\right\}$ ▷ compute the probability distribution of final plans according to the third step of the BLLL (scaling the utility function with the related max value)

16: $\mathcal{M}_i(t) =$ random_selection$\left(\{\mathcal{M}_i^* \cup \mathcal{M}_i^l\}, P_{\text{final}}\right)$ ▷ 2nd random selection (final plan proposal)

17: $k =$ random_selection$(\mathbb{D})$ ▷ random selection of the next drone k for proposing

18: drone_send_message$\left(k, \mathcal{M}_i(t), \mathcal{M}_{-i}(t-1)\right)$ ▷ send the selected proposal and the last other proposals to the drone k

19: **return** $\mathcal{M}_i(t)$

20: **end**

21 **end**

FIGURE 6. Pseudo code of the first solution for the coordination algorithm.

calculating the probabilities in (9) and in (10) in a practical implementation. For this purpose, a normalization method, derived from the robust scaler technique [57], is used. This method replaces median, second and third quartile of the robust scaler with parametric percentiles in order to further reward the higher utilities without resorting to too small values for β. However, a limitation of the solution in Fig. 6 is represented by the rare selection of ancestor plans, which could not allow a sufficient diversity in the exploration strategy over the set $\mathbb{M}_i$, e.g., to escape from local equilibria caused by previous selections in the sequence. Therefore, we have also designed a second solution for the coordination algorithm, which modifies the third point of the BLLL's structure by increasing the probability to select an ancestor mission plan acting on the temperature τ. This second solution has been explicitly implemented for the case $w = 1$ for computational reasons, even if it is theoretically applicable for any value of w.

In detail, in this second solution, the ith agent will select the new mission plan (modifying the line 15 of the algorithm in Fig. 6) according to the following probabilities,

$$P\left(\mathcal{M}_i(t) = \mathcal{M}_i^*\right) = \frac{e^{\frac{1}{\tau}u_i(\mathcal{M}_i^*, \mathcal{M}_{-i}(t-1))}}{\sum(\cdot)}, \tag{21}$$

$$P\left(\mathcal{M}_i(t) = \mathcal{M}_i(t-1)\right) = \frac{e^{\frac{1}{\tau}u_i(\mathcal{M}_i(t-1), \mathcal{M}_{-i}(t-1))}}{\sum(\cdot)}, \tag{22}$$

$$P\left(\mathcal{M}_i(t) = \mathcal{M}_i^{[-1]}(t-1)\right) = \frac{e^{\frac{1}{\tau}u_i\left(\mathcal{M}_i^{[-1]}(t-1), \mathcal{M}_{-i}(t-1)\right)}}{\sum(\cdot)}, \tag{23}$$

where,

$$\sum(\cdot) \triangleq e^{\frac{1}{\tau}u_i(\mathcal{M}_i^*, \mathcal{M}_{-i}(t-1))} + e^{\frac{1}{\tau}u_i(\mathcal{M}_i(t), \mathcal{M}_{-i}(t-1))} + e^{\frac{1}{\tau}u_i\left(\mathcal{M}_i^{[-1]}(t-1), \mathcal{M}_{-i}(t-1)\right.}. \tag{24}$$

Strictly speaking, the demonstrations of reachability and reversibility properties are valid for both algorithms. Instead, the stochastic stability of G maximizers is formally valid only for the first solution, since this adopt the same "double-selection" structure in (9) and (10) of the basic BLLL for the selection of the new action. Instead, the second solution exhibits the aforementioned "triple-selection" structure, which requires further analysis to evaluate the stochastic stability of G maximizers. The next section reports detailed experimental analysis, also to show the performance difference between the first and the second solution.

V. EXPERIMENTAL ANALYSIS

The proposed solutions for the problem in (6) were tested by means of model-in-the-loop simulations. The Mesa framework [58] was used for the coding and the simulation of the agent-based models underlying the planner and the drone team. It is a modular Python-based framework for building, analysing, simulating and visualizing agent-based models. Our previous work [16] already reported some experimental results with the support of Mesa framework to confirm the effectiveness of a preliminary version of the learning design. These results especially concerned a comparison with a centralized coordination method (exploiting K-means algorithm), by assessing an efficiency metric as the ratio between the ECI values of our solution and the centralized one. The previous work demonstrated that the efficiency increases with the number of targets, thus the game-theoretic coordination mechanism is scalable with respect to the complexity of scenario (i.e., the number of waypoints in the ROI).

For this work, some specific simulation campaigns were set with:

- a mission with a duration $\mathcal{T}$ equal to 10 '000 s (2.78 hours);
- a 5-drone team with an average cruise velocity equal to 1 m/s for each drone and with enough endurance to cover the duration $\mathcal{T}$;
- a ROI having a surface area equal to 400 m × 400 m;
- a free-space ROI (i.e., with no-fly zones and obstacles);
- a home (i.e., the starting point of the drones) included in the ROI and with the same positions for all the drones in the team;
- a variable number of sensing waypoints located in the ROI;
- homogeneous targets, i.e., with the same priority and the same time needed for the inspection sensing (assumed to be null).

The conditions about free-space ROI and homogeneous target are without loss of generality for the proposed solutions and are adopted just to facilitate the experimental analysis. For example, the impact of possible obstacles may be easily processed by computing the increase of ECI variations for the edges between sensing waypoints which pass through the obstacles, and by adding the proper intermediate waypoints in the mission plans.

As example, Fig. 7 shows the mission plans for each drone provided by the first proposed solution at the end of a simulation run, for two scenarios characterized respectively by 50 and 100 tasks with the home centered in the ROI.

The sample average of the minimum ECI (ECI_{min}) and, equivalently, of the maximum mission utility $G(\mathcal{M})$ (G_{max}), were calculated as metrics to evaluate the performance of the proposed solutions. As additional reference metrics, we considered: (i) the number of assigned tasks in the joint plan $N(t) = |\mathcal{M}(t)|$, that is the sum of cardinalities of $\mathcal{M}_i$, i.e., $N(t) = \sum_{i=1}^{N_{\text{d}}} |\mathcal{M}_i(t)|$; (ii) the Convergence Step (CS) as the number of steps after which the coordination algorithm obtains the 99% of the minimum value of the ECI of the joint mission plan in the learning sequence. Each step is also named learning step or negotiation step, and we use the term 'negotiation' to denote the iterative learning process to reach an equilibrium in the joint plan selection.

TABLE 1. Simulations results of the first solution.

S	w	β	τ	$\overline{\text{CS}}$	ECI_{min}		
					μ	σ	CI @ 95%
1	1	$\to 0$	0.01	67.1	15206	1568	[15068, 15343]
2	1	$\to 0$	0.05	73.3	15174	1561	[15037, 15311]
3	1	$\to 0$	0.1	79.5	15158	1629	[15015, 15301]
4	1	$\to 0$	0.5	103.6	15169	1561	[15032, 15306]
5	1	$\to 0$	1	114.1	15199	1575	[15060, 15337]
6	1	$\to 0$	5	119.6	15303	1508	[15170, 15435]
7	1	$\to 0$	10	122.7	15158	1500	[15027, 15290]
8	1	$\to 0$	100	124.1	15114	1571	[14976, 15252]
9	1	0.01	0.01	69.0	15195	1588	[15055, 15334]
10	1	0.05	0.01	68.9	15438	1662	[15292, 15583]
11	1	0.1	0.01	69.4	15704	1558	[15567, 15840]
12	1	0.5	0.01	74.8	24639	2294	[24438, 24840]
13	1	1	0.01	92.6	34455	3389	[34158, 34752]
14	2	$\to 0$	0.01	36.8	14545	1300	[14431, 14658]

The first simulation campaign aimed at studying the effectiveness of the first solution, also by preliminarily assessing how the variation of its main parameters influences the statistics of the performance of the algorithm. In such campaign, the home position was centered in the ROI and 50 sensing tasks were considered. The first solution was analyzed using the values 100 and 90 for the percentiles α and γ, and assessing the influence of w, β and τ in three steps. The influence of the temperature τ was firstly assessed by using $w = 1$ and by adopting the condition $\beta \to 0$, which corresponds to a best response learning policy (i.e., the plan with the highest utility is deterministically selected) for the first random selection (line 13 of the algorithm in Fig. 6). Afterwards, the influence of the temperature β was assessed by using $w = 1$ and the optimal setting of τ (i.e., the one minimizing ECI_{min}) as identified in the previous step. Lastly, the influence of the planning window w was assessed, using the optimal settings of β and τ previously identified. The evaluation was performed by simulating 500 random scenarios (i.e., random spatial distributions of the waypoints associated to the 50 sensing tasks).

The results are shown in Table 1, where: S indicates the identifier of the simulation set (i.e., the aggregation of random scenarios); $\overline{\text{CS}}$ represents the sample mean of the CS metric; μ, σ and CI respectively indicate the sample mean, the sample standard deviation and the confidence interval of ECI_{min}.

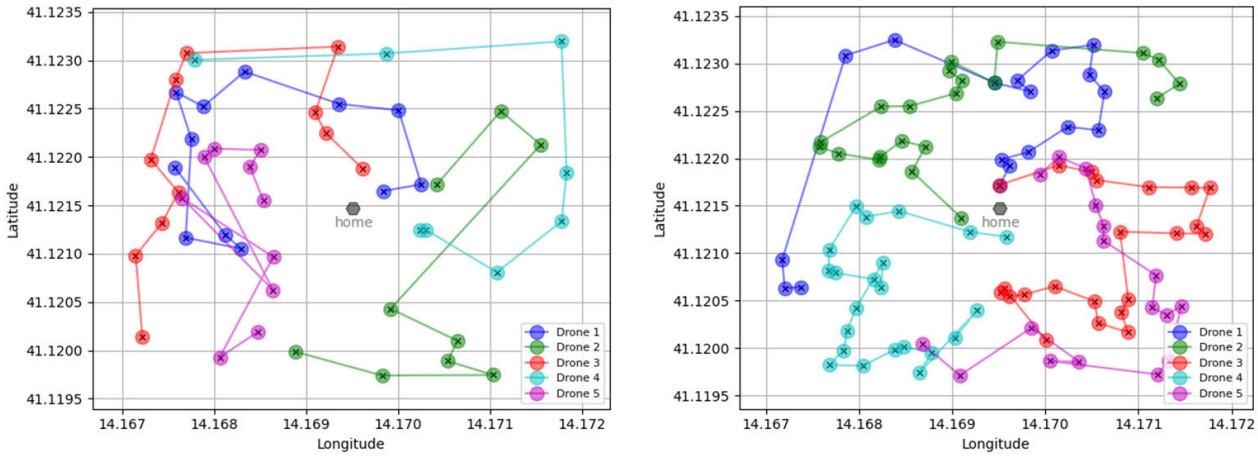


FIGURE 7. Mission plans assigned to the drone team provided by the first solution for the coordination algorithm considering a first scenario with 50 sensing tasks (left) and a second scenario with 100 sensing tasks (right).

In regard to the influence of τ, such table highlights that: (i) variations of τ (simulation sets 1-8) do not produce remarkable changes of the statistics of ECI_{min} , whereas $\overline{\text{CS}}$ increases with τ (an explanation for this is subsequently provided); (ii) the best setting for τ is 0.01 within best response policy (simulation sets 1-8). In regard to β, this does not significantly affect the statistics of ECI_{min} values in the range $]0, 0.1]$, whereas it produces worse average values of ECI_{min} for $\beta > 0.1$. This is because, as β increases, the coordination algorithm allows the drones to add tasks to their mission plans in an increasingly random manner. Also note that $\overline{\text{CS}}$ increases with β (an explanation for this is subsequently provided). Lastly, both statistics of ECI_{min} and CS improve as w is increased, as evident for the last simulation set in Table 1. Indeed, greater values of w allow an enhanced research for the agent i in the constrained set $\mathbb{C}_i^w$ $(\mathcal{M}_i(t))$, since this set contains more plans if w is increased. On the other hand, larger planning windows imply a greater computational load for the online generation of individual mission plans.

The above results also confirm that, for the first random selection (line 13 of the algorithm in Fig. 6), a simple weighted distribution (i.e., without exponential functions) would not be suitable to select $\mathcal{M}_i^*$. Indeed, better performance is achieved for lower values of β (i.e., for near best response policy), resulting in an exponential weighing providing larger increments for the related cumulative distribution function at the mission plans with the highest utility values. To the contrary, greater values of β tend towards a uniform distribution.

Fig. 8 shows other views of the results of the first campaign to better understand their implications, related to the time evolution of negotiation. In detail, such evolution is generally characterized by the following phases:

- First negotiation phase - This is the initial phase of the negotiation evolution, which is distinguished by the presence of tasks that are not allocated in any individual plans, i.e., $\mathbb{T}_{\text{u}} \neq \emptyset$. Here, at each time step the coordination algorithm allows the enabled agent to select either its previous plan or a new plan. In the former case, the joint plan and the total mission utility do not change with respect to the previous step, so the negotiation step is also named a ''stalemate'' step. In the latter case, the algorithm tends to lean towards not allocated tasks, i.e., to select other tasks in $\mathbb{T}_u$. This phase on average has a quick growth in the mission utility, as a consequence of (18).
- Second negotiation phase - Once all the tasks have been allocated to at least one drone ($\mathbb{T}_{\text{u}} = \emptyset$), a second phase starts, wherein agents may compete for tasks already allocated. Here, the mission utility on average has a slower growth with respect to the first phase, as a consequence of (19).
- Equilibrium phase - The second phase continues until an equilibrium is reached, which is distinguished by a stable joint plan that is not perturbed by the coordination algorithm.

The aforementioned phases are evident in the charts in Fig. 8, which all depict the time evolution of the mission utility for given simulation runs related to different combinations of the temperatures β and τ, with $w = 1$.

Fig. 8 (bottom-left) shows a clear example of the influence of the temperature τ, whose variation does not imply remarkable changes of the equilibrium value of $G(\mathcal{M}(t))$. Indeed, τ only influences the number of stalemate steps occurred during the second negotiation phase, resulting in a mere increase of the value of the CS. Instead, Fig. 8 (bottomright) highlights the reduction of the equilibrium values of $G(\mathcal{M}(t))$ and the increase of the CS as β increases. In detail, the value of the CS gets larger because the number of stalemate steps increase and the second negotiation phase takes more steps to converge, due to the greater randomness in the selection of new plans.

Lastly, Fig. 9 illustrates a comparison between the simulation sets 1 and 14, characterized by the same settings except

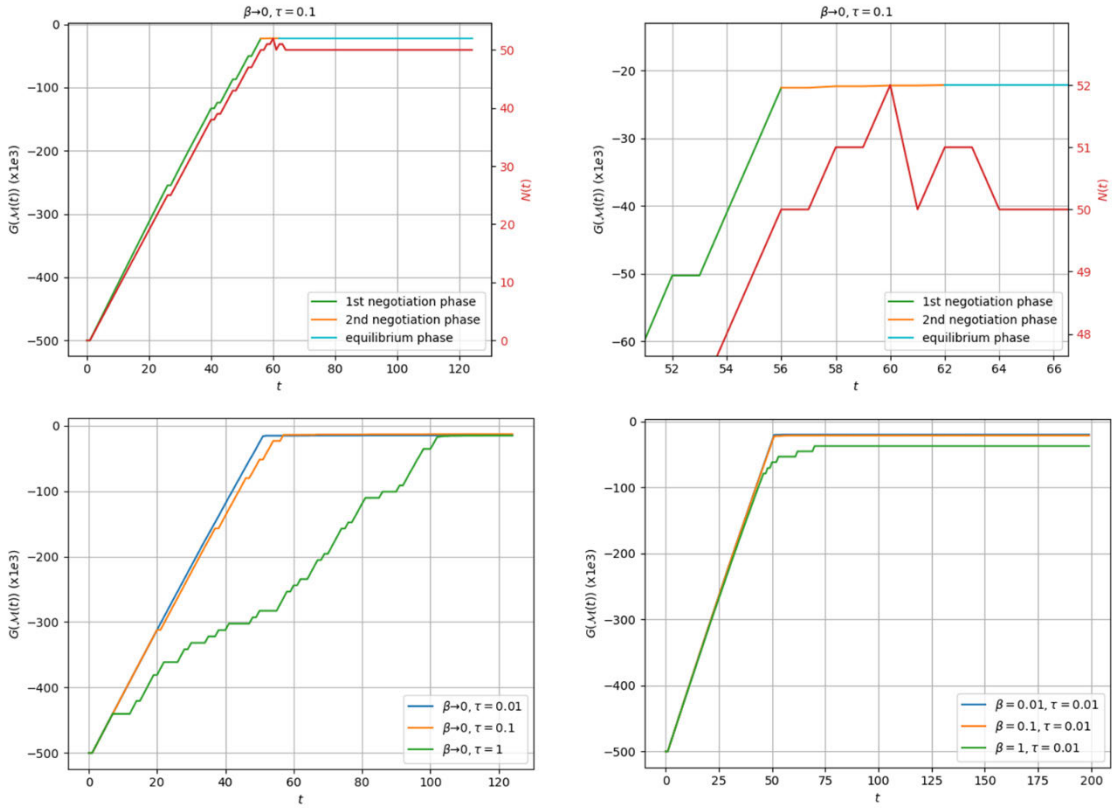


FIGURE 8. (top-left) Mission utility and number of sensing tasks selected by the drones for a given scenario with the configuration of the simulation set 3; (top-right) Detail of the mission utility shown in (top-left) illustrating the transition between the negotiation phases; (bottom-left) and (bottom-right) Mission utility for different configuration of temperatures β and τ. All the data shown in the figures refer to the first solution for the coordination algorithm with $w = 1$.

for w. Such figure demonstrates how the statistics of ECI_{min} and G_{max} improve as w increases by reporting a histogram of the occurrences of G_{max} respectively for $w = 1$ and $w = 2$.

Afterwards, the second simulation campaign aimed at studying the effectiveness of the second solution, with the same approach of the first campaign. The solution was assessed using the same 500 random scenarios and the same values for the percentiles α and γ of the first campaign. In this case, only the influence of β and τ was assessed, whereas a constant planning window $w = 1$ was adopted.

With the second solution, the negotiation is expected to not reach an equilibrium because further randomness is introduced in the second random selection (line 16 in Fig. 6) allowing a drone to deliberate an ancestor plan with a not null probability. Therefore, the value of ECI_{min} is a function of the number of time steps t_{max} granted to the coordination. Thus, also the influence of such number of time steps should be assessed in this case. In detail, we considered two different values of t_{max}, respectively 600 and 1000 steps, to find the best combination of t_{max} and τ in the case $\beta \to 0$. Then, other simulations were carried out considering different values of β to assess its influence. The obtained results are shown in Table 2.

In the simulation sets 1-5 ($t_{\text{max}} = 600$ and $\beta \to 0$), a small improvement in performance was observed for small values

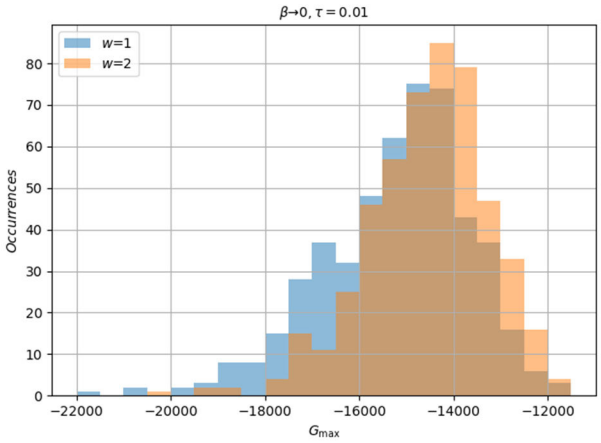


FIGURE 9. Comparison between performances obtainable with two different values (respectively 1 and 2) for w using the first solution for the coordination algorithm.

of τ (≤ 0.1) with respect to the average value of ECI_{min} provided by the first solution (considering $w = 1$). However, a significant increase was detected for the value of $\overline{\text{CS}}$. Moreover, a worsening of the statistics of ECI_{min} was observed for $\tau > 0.1$, in contrast to the first solution. Instead, in simulation sets 6-10 ($t_{\text{max}} = 1000$ and $\beta \to 0$),

TABLE 2. Simulations of the second solution.

S	β	t_{max}	τ	$\overline{CS}$	ECI_{min}		
					μ	σ	CI @ 95%
1	$\to 0$	600	0.01	87.5	15088	1563	[14951, 15225]
2	$\to 0$	600	0.05	156.2	15059	1585	[14920, 15198]
3	$\to 0$	600	0.1	168.3	15086	1553	[14950, 15222]
4	$\to 0$	600	0.5	274.9	16274	1777	[16118, 16430]
5	$\to 0$	600	1	376.9	17814	5936	[17293, 18334]
6	$\to 0$	1000	0.01	92.5	15073	1572	[14935, 15211]
7	$\to 0$	1000	0.05	204.7	15039	1580	[14900, 15177]
8	$\to 0$	1000	0.1	219.6	15119	1578	[14810, 15428]
9	$\to 0$	1000	0.5	289.1	16354	1834	[16193, 16515]
10	$\to 0$	1000	1	423.2	17044	1974	[16871, 17217]
11	0.01	600	0.05	133.4	15176	1616	[15035, 15318]
12	0.05	600	0.05	136.6	15410	1645	[15266, 15554]
13	0.1	600	0.05	143.3	15664	1575	[15526, 15802]
14	0.5	600	0.05	173.2	24163	2281	[23963, 24363]

no significant differences for the statistics of ECI_{min} were noticed with respect to simulation sets $1-5$. To conclude: (i) the optimal setting of τ is 0.05 and (ii) the increase t_{max} does not produce notable improvement. Moreover, using the settings for $\tau = 0.05$ and $t_{max} = 600$, no improvements were observed for $\beta < 0.05$ in simulation sets $11-14$ with respect to the previous simulation sets, whereas some worsening was observed for $\beta > 0.5$. This influence of β is similar to the results of the assessment of β obtained for the first solution.

Fig. 10 and Fig. 11 provide a temporal view of the above results for a sample simulation scenario, confirming the absence of the equilibrium phase. Indeed, a persistent variation of $G(\mathcal{M})$ and $N(t)$ is evident in Fig. 11.

An additional comparison between the first and the second solution is shown by means of the histogram in Fig. 12, where the occurrences of the values of mission utility are reported with the same settings for $\beta \to 0$, $\tau = 0.05$ and $w = 1$ for both solutions, and with $t_{max} = 600$ for the second solution. The histogram experimentally confirms the slight improvement introduced by the second solution for the average values of G_{max}. As mentioned in section IV.C, this experimental analysis shall be complemented with a formal analysis of the stochastic stability of the maximizers of the mission utility function.

The third simulation campaign aimed at assessing the influence of the distribution of sensing tasks on the performance of the coordination algorithm. For this purpose, we used the following configuration for the parameters of the first solution: (i) the values 100 and 90 for the percentiles α and γ, (ii) $\beta \to 0$, (iii) $\tau = 0.01$ and (iv) 100 random scenarios for each simulation set, where the settings for β and τ

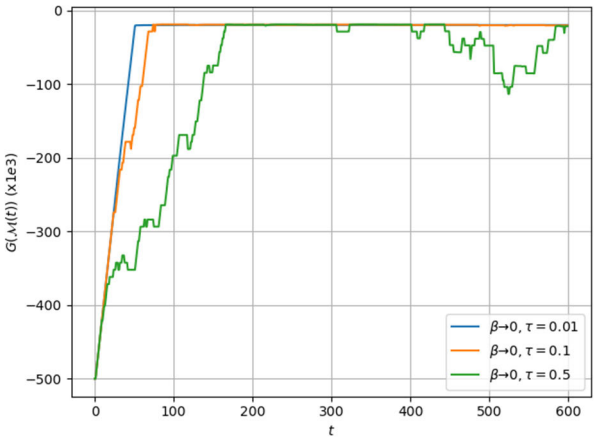


FIGURE 10. Mission utility for different configuration of τ with $\beta \to 0$, obtained with the second solution for the coordination algorithm.

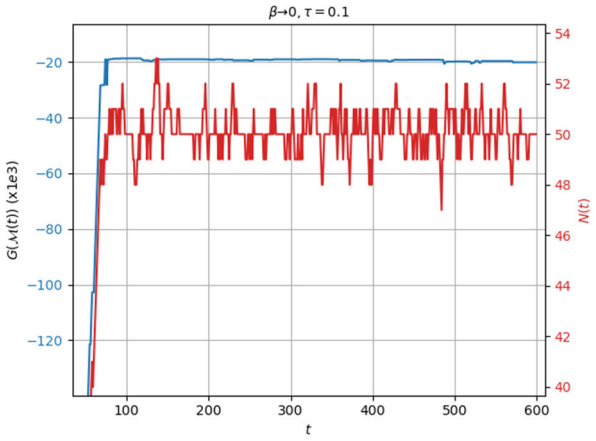


FIGURE 11. Detail of the mission utility and the total sensing tasks selected by the drones for a given scenario with the configuration defined for the simulation set 3 in Table 2. The data shown refer to the second solution for the coordination algorithm.

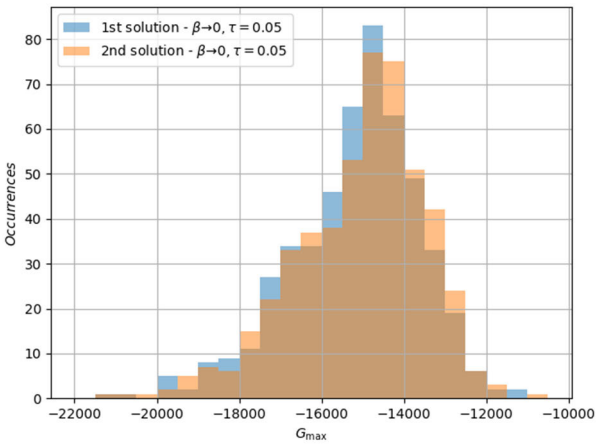


FIGURE 12. Comparison between the maximum mission utility provided by the first and the second solutions of for the coordination algorithm considering $\beta \to 0$ and $\tau = 0.05$.

represent the best trade-off for the first solution, as identified in the first simulation campaign. The analysis was carried out by varying: (i) $N_t \in \{50, 100, 150, 200, 250, 300\}$; (ii) the

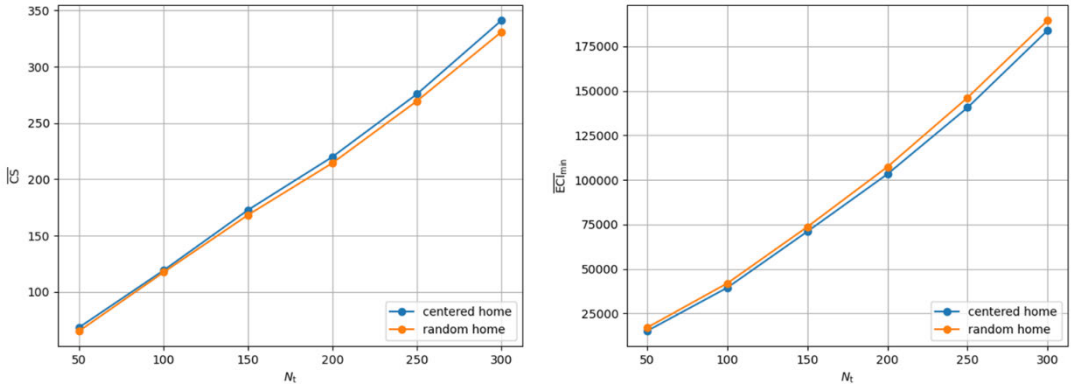


FIGURE 13. (left) Average convergence step vs. number of sensing tasks considered in the simulation sets; (right) Sample mean of ECI_{min} vs. number of sensing tasks considered in the simulation sets. The data shown in the figures refer to the first solution for the coordination algorithm, considering the settings $\beta \to 0$, $\tau = 0.01$ and $w = 1$.

TABLE 3. Simulations of the third campaign.

S	HP	N_t	w	$\overline{CS}$	ECI_{min}		
					μ	σ	CI @ 95%
1	centered	50	1	68.6	15249	1517	[14950, 15547]
2	centered	50	2	35.7	14447	1257	[14201, 14694]
3	centered	100	1	119.2	39528	3260	[38889, 40167]
4	centered	100	2	64.5	38009	3005	[37420, 38597]
5	centered	150	1	172.8	71074	6028	[69892, 72255]
6	centered	200	1	220.0	103316	7768	[101794, 104839]
7	centered	250	1	275.7	140650	8394	[139005, 142295]
8	centered	300	1	341.0	184019	10167	[182027, 186012]
9	random	50	1	65.4	17129	1779	[16780, 17477]
10	random	50	2	38.6	16383	1616	[16066, 16700]
11	random	100	1	117.5	41979	3816	[41231, 42727]
12	random	100	2	61.2	40731	3032	[40137, 41325]
13	random	150	1	168.4	73694	5657	[72585, 74802]
14	random	200	1	214.5	107381	7784	[105855, 108907]
15	random	250	1	269.5	146236	8689	[144533, 147939]
16	random	300	1	330.8	189516	10601	[187438, 191594]

home position, which might be fixed and centered in the ROI, or it might be randomly located (with a uniform distribution) in the ROI. For the planning window, the setting $w = 1$ was used for all the values of N_t, whereas the setting $w = 2$ was also used for $N_t = 50$ and $N_t = 100$. The results are shown in Table 3, where HP indicates the home position (centered or random).

Based on the results in Table 3, Fig. 13 (left) and (right) respectively show the average value of the CS and of ECI_{min} as functions of N_t for both random and centered home positions, with $w = 1$. In detail, Fig. 13 (left) highlights that the convergence step increases almost linearly with N_t for each configuration, confirming the scalability of the proposed coordination algorithms.

VI. CONCLUSION

This work reports the design of a dynamic and decentralized mission planner for a drone team performing autonomous and cooperative spatio-temporal sensing. It applies the learning-ingames framework for the coordination of the team, introducing ad-hoc variants of the binary log-linear learning. Also, the work provides a mathematical analysis of the reachability and reversibility properties for the designed learning-in-games approach. Lastly, a detailed experimental analysis is presented, which allows for a preliminary tune of some learning parameters.

Future work entails a systematic optimization of the learning parameters with respect to the mission scenario and a trade-off with other reinforcement-learning approaches. For example, Agent-based Evolutionary Search (AES) may be applied for the optimization strategy of the learning parameters, similarly to other multi-agent applications [59]. Moreover, a formal analysis shall be performed to evaluate the stochastic stability of the maximizers of the mission utility function for the second coordination solution, considering possibly additional frameworks for the stability analysis of learning in games [60]. In case of spatio-temporal sensing for security-driven applications, the coordination mechanism shall be enriched to consider advanced and intelligent behaviours of the reference threat, e.g., by means of Stackelberg security games. Lastly, future work shall consider actual flight tests to further verify the performance of the proposed mission planner in a real environment.

APPENDIX A
PROOF OF THEOREM 1—REACHABILITY

We firstly consider the case where $\mathcal{M}_i^m$ is an ancestor or a descendant of $\mathcal{M}_i^0$. If the plans $\mathcal{M}_i^0$ and $\mathcal{M}_i^m$ differ in r

tasks with $r \leq w$, the proof is trivial, because the property is verified as per construction of $\mathbb{C}_i^w$. Otherwise ($r > w$):

- if $\mathcal{M}_i^m$ is a descendant of $\mathcal{M}_i^0$, we can apply a sequence of ceil (r/w) steps to add the r differing tasks to the previous plan, starting from $\mathcal{M}_i^0$, to obtain $\mathcal{M}_i^m$;
- if $\mathcal{M}_i^m$ is a descendant of $\mathcal{M}_i^0$, we can apply a sequence of ceil (r/w) steps to subtract the r differing tasks from the previous plan, starting from $\mathcal{M}_i^0$, to obtain $\mathcal{M}_i^m$.

Instead, if $\mathcal{M}_i^m$ is not an ancestor or a descendant of $\mathcal{M}_i^0$, we can apply a first sequence of plans, which starts from $\mathcal{M}_i^0$ and subtracts tasks from the previous plan of each sequence step, until it reaches the empty plan. Then, we can apply a second sequence of plans, which starts from the empty sequence and adds the tasks of $\mathcal{M}_i^m$ to the previous mission of each sequence, until it reaches the fully $\mathcal{M}_i^m$. Merging the two sequences, the property is proved also in this case.

APPENDIX B
PROOF OF THEOREM 2—REVERSIBILITY

As per construction of $\mathbb{C}_i^w$, if $\mathcal{M}_i^0 \in \mathbb{C}_i^w\left(\mathcal{M}_i^1\right)$, then $\mathcal{M}_i^0$ can be: (i) a j th order descendant of $\mathcal{M}_i^1$ with $j \leq w$; (ii) a j th order ancestor of P_i^1 with $j \leq w$; or (iii) equal to $\mathcal{M}_i^1$.

In the case (i), since $\mathbb{C}_i^w\left(\mathcal{M}_i^0\right)$ includes all the ancestors of $\mathcal{M}_i^0$ up to (at least) the order j, therefore it includes also $\mathcal{M}_i^1$.

In the case (ii), since $\mathbb{C}_i^w\left(\mathcal{M}_i^0\right)$ includes all the descendant of $\mathcal{M}_i^0$ up to (at least) the order j, therefore it includes also $\mathcal{M}_i^1$.

In the case (iii), the proof of the property is trivial.

REFERENCES

[1] NIST. (Oct. 2008). *Autonomy Levels for Unmanned Systems (ALFUS) Framework–Volume I: Terminology*. Hui-Min Huang National Institute of Standards & Technology, Gaithersburg, MD, USA, NIST Special Publication 1011-I-2.0. [Online]. Available: https://www.nist.gov/system/files/documents/el/isd/ks/NISTSP_1011-I-2-0.pdf

[2] E. Gat, "Autonomy software verification and validation might not be as hard as it seems," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, vol. 5, Mar. 2004, pp. 3123–3128, doi: 10.1109/AERO.2004.1368117.

[3] S. E. Kolitz and R. M. Beaton, "Overall system concepts in mission planning," in *New Advances in Mission Planning and Rehearsal Systems, AGARD Lecture Series*, vol. 192. Neuilly Sur Seine, France: Advisory Group for Aerospace Research & Development, 1993. [Online]. Available: https://www.sto.nato.int/publications/AGARD/AGARD-LS-192/AGARD-LS-192.pdf

[4] C. Tabasso, C. Kielas-Jensen, V. Cichella, S. Manyam, D. W. Casbeer, and I. Weintraub, "Continuous monitoring of a path-constrained moving target by multiple unmanned aerial vehicles," *J. Guid., Control, Dyn.*, vol. 45, no. 4, pp. 704–713, Apr. 2022, doi: 10.2514/1.g006043.

[5] C. Tabasso, V. Cichella, S. B. Mehdi, T. Marinho, and N. Hovakimyan, "Time coordination and collision avoidance using leader-follower strategies in multi-vehicle missions," *Robotics*, vol. 10, no. 1, p. 34, Feb. 2021, doi: 10.3390/robotics10010034.

[6] V. Cichella, "Cooperative autonomous systems: Motion planning and coordinated tracking control for multi-vehicle missions. dissertation," Ph.D. dissertation, Mech. Eng., Univ. Illinois Urbana-Champaign, Urbana, IL, USA, 2018. [Online]. Available: https://core.ac.uk/download/pdf/161953638.pdf

[7] W. Truszkowski, H. L. Hallock, C. Rouff, J. Karlin, J. Rash, M. Hinchey, and R. Sterritt, "Cooperative autonomy," in *Autonomous and Autonomic Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems* (NASA Monographs in Systems and Software Engineering), London, U.K.: Springer, 2009, doi: 10.1007/b105417_7.

[8] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," in *Proc. AAAI Fall Symp.*, 2007, pp. 1–6. [Online]. Available: https://cdn.aaai.org/Symposia/Fall/2007/FS-07-06/FS07-06-002.pdf

[9] J. K. Verma and V. Ranga, "Multi-robot coordination analysis, taxonomy, challenges and future scope," *J. Intell. Robotic Syst.*, vol. 102, p. 10, Apr. 2021, doi: 10.1007/s10846-021-01378-2.

[10] L. Beaudoin, A. Gademer, L. Avanthey, V. Germain, and V. Vittori, "Potential threats of UAS swarms and the countermeasure's need," in *Proc. Eur. Conf. Inf. Warfare Secur. (ECIW)*, Tallinn, Estonia, Jul. 2011, pp. 24–30. [Online]. Available: https://hal.science/hal-01132236/document

[11] V. U. Castrillo, A. Manco, D. Pascarella, and G. Gigante, "A review of counter-UAS technologies for cooperative defensive teams of drones," *Drones*, vol. 6, no. 3, p. 65, Mar. 2022, doi: 10.3390/drones6030065.

[12] X. Wang, Y. Cao, M. Ding, X. Wang, W. Yu, and B. Guo, "Research progress in modeling and evaluation of cooperative operation system-of-systems for manned-unmanned aerial vehicles," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 39, no. 4, pp. 6–31, Apr. 2024, doi: 10.1109/MAES.2023.3347504.

[13] J. Song, K. Zhao, and Y. Liu, "Survey on mission planning of multiple unmanned aerial vehicles," *Aerospace*, vol. 10, no. 3, p. 208, Feb. 2023, doi: 10.3390/aerospace10030208.

[14] C. Qin and E. Pournaras, "Coordination of drones at scale: Decentralized energy-aware swarm intelligence for spatio-temporal sensing," *Transp. Res. C, Emerg. Technol.*, vol. 157, Dec. 2023, Art. no. 104387, doi: 10.1016/j.trc.2023.104387.

[15] G. Gigante, D. Pascarella, S. Luongo, C. Di Benedetto, A. Vozella, and G. Persechino, "Game-theoretic approach for the optimal configuration computing of an interoperable fleet of unmanned vehicles," *Expert Syst.*, vol. 35, Oct. 2018, Art. no. e12293, doi: 10.1111/exsy.12293.

[16] V. U. Castrillo, I. Iudice, D. Pascarella, G. Pigliasco, and A. Vozella, "Game-theoretic mission planning of drone teams in autonomous detection and recognition," in *Proc. IEEE Int. Workshop Technol. Defense Secur. (TechDefense)*, Rome, Italy, Nov. 2023, pp. 197–202, doi: 10.1109/TechDefense59795.2023.10380873.

[17] W. Ren and Y. Cao, "Overview of recent research in distributed multi-agent coordination," in *Distributed Coordination of Multi-Agent Networks: Emergent Problems, Models, and Issues*. London, U.K.: Springer, 2011, pp. 23–41, doi: 10.1007/978-0-85729-169-1_2.

[18] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28573–28593, 2018, doi: 10.1109/ACCESS.2018.2831228.

[19] I. Maza, J. Capitán, L. Merino, and A. Ollero, "Multi-UAV cooperation," in *Encyclopedia of Aerospace Engineering*. Hoboken, NJ, USA: Wiley, 2015, doi: 10.1002/9780470686652.eae1130.

[20] V. Hassija, V. Chamola, A. Agrawal, A. Goyal, N. C. Luong, D. Niyato, F. R. Yu, and M. Guizani, "Fast, reliable, and secure drone communication: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2802–2832, 4th Quart., 2021, doi: 10.1109/COMST.2021.3097916.

[21] Y. Mekdad, A. Aris, L. Babun, A. E. Fergougui, M. Conti, R. Lazzeretti, and A. S. Uluagac, "A survey on security and privacy issues of UAVs," *Comput. Netw.*, vol. 224, Apr. 2023, Art. no. 109626, doi: 10.1016/j.comnet.2023.109626.

[22] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3417–3442, 4th Quart., 2019, doi: 10.1109/COMST.2019.2906228.

[23] S. D. Apostolidis, P. C. Kapoutsis, A. C. Kapoutsis, and E. B. Kosmatopoulos, "Cooperative multi-UAV coverage mission planning platform for remote sensing applications," *Auton. Robots*, vol. 46, no. 2, pp. 373–400, Feb. 2022, doi: 10.1007/s10514-021-10028-3.

[24] Á. Calvo, G. Silano, and J. Capitán, "Mission planning and execution in heterogeneous teams of aerial robots supporting power line inspection operations," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Dubrovnik, Croatia, Jun. 2022, pp. 1644–1649, doi: 10.1109/ICUAS54217.2022.9836234.

[25] S. Poudel and S. Moh, "Task assignment algorithms for unmanned aerial vehicle networks: A comprehensive survey," *Veh. Commun.*, vol. 35, Jun. 2022, Art. no. 100469, doi: 10.1016/j.vehcom.2022.100469.

[26] G. M. Skaltsis, H. S. Shin, and A. Tsourdos, "A review of task allocation methods for UAVs," *J. Intell. Robotic Syst.*, vol. 109, p. 76, Nov. 2023, doi: 10.1007/s10846-023-02011-0.

[27] H. Yan, W. Zhao, C. Chen, Y. You, X. Gao, D. Zhang, W. Cao, and W. Bao, "MCTA: Multi-UAV collaborative target allocation to monitor targets with dynamic importance," in *Proc. 6th Int. Conf. Big Data Inf. Anal. (BigDIA)*, Shenzhen, China, Dec. 2020, pp. 50–57, doi: 10.1109/BigDIA51454.2020.00017.

[28] J. Chen, X. Qing, F. Ye, K. Xiao, K. You, and Q. Sun, "Consensus-based bundle algorithm with local replanning for heterogeneous multi-UAV system in the time-sensitive and dynamic environment," *J. Supercomput.*, vol. 78, no. 2, pp. 1712–1740, Feb. 2022, doi: 10.1007/s11227-021-03940-z.

[29] A. Liekna, E. Lavendelis, and A. Grabovskis, "Experimental analysis of contract net protocol in multi-robot task allocation," *Appl. Comput. Syst.*, vol. 13, no. 1, pp. 6–14, 2013, doi: 10.2478/v10312-012-0001-7.

[30] J. J. Roldán, J. Del Cerro, and A. Barrientos, "Should we compete or should we cooperate? Applying game theory to task allocation in drone swarms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Madrid, Spain, Oct. 2018, pp. 5366–5371, doi: 10.1109/IROS.2018.8594145.

[31] G. Attenni, V. Arrigoni, N. Bartolini, and G. Maselli, "Drone-based delivery systems: A survey on route planning," *IEEE Access*, vol. 11, pp. 123476–123504, 2023, doi: 10.1109/ACCESS.2023.3329195.

[32] A. M. Raivi, S. M. A. Huda, M. M. Alam, and S. Moh, "Drone routing for drone-based delivery systems: A review of trajectory planning, charging, and security," *Sensors*, vol. 23, no. 3, p. 1463, Jan. 2023, doi: 10.3390/s23031463.

[33] F. Aljalaud, H. Kurdi, and K. Youcef-Toumi, "Bio-inspired multi-UAV path planning heuristics: A review," *Mathematics*, vol. 11, no. 10, p. 2356, May 2023, doi: 10.3390/math11102356.

[34] S. Lin, A. Liu, J. Wang, and X. Kong, "A review of path-planning approaches for multiple mobile robots," *Machines*, vol. 10, no. 9, p. 773, Sep. 2022, doi: 10.3390/machines10090773.

[35] G. Macrina, L. Di Puglia Pugliese, F. Guerriero, and G. Laporte, "Drone-aided routing: A literature review," *Transp. Res. C, Emerg. Technol.*, vol. 120, Nov. 2020, Art. no. 102762, doi: 10.1016/j.trc.2020.102762.

[36] C. Cheng, Y. Adulyasak, and L. Rousseau, "Formulations and exact algorithms for drone routing problem," CIRRELT, Montreal, QC, Canada, Tech. Rep., Jul. 2018. [Online]. Available: https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2018-31.pdf

[37] S. Cavani, M. Iori, and R. Roberti, "Exact methods for the traveling salesman problem with multiple drones," *Transp. Res. C, Emerg. Technol.*, vol. 130, Sep. 2021, Art. no. 103280, doi: 10.1016/j.trc.2021.103280.

[38] C. C. Murray and R. Raj, "The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones," *Transp. Res. C, Emerg. Technol.*, vol. 110, pp. 368–398, Jan. 2020, doi: 10.1016/j.trc.2019.11.003.

[39] Z. Wang and J. Sheu, "Vehicle routing problem with drones," *Transp. Res. B, Methodol.*, vol. 122, pp. 350–364, Apr. 2019, doi: 10.1016/j.trb.2019.03.005.

[40] J. M. Leon-Blanco, P. L. Gonzalez-R, J. L. Andrade-Pineda, D. Canca, and M. Calle, "A multi-agent approach to the truck multi-drone routing problem," *Expert Syst. Appl.*, vol. 195, Jun. 2022, Art. no. 116604, doi: 10.1016/j.eswa.2022.116604.

[41] X. Kong, Y. Zhou, Z. Li, and S. Wang, "Multi-UAV simultaneous target assignment and path planning based on deep reinforcement learning in dynamic multiple obstacles environments," *Frontiers Neurorobot.*, vol. 17, Jan. 2024, Art. no. 1302898, doi: fnbot.2023.1302898.

[42] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multi-agent pickup and delivery," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5816–5823, Jul. 2021, doi: 10.1109/LRA.2021.3074883.

[43] J. Ni, G. Tang, Z. Mo, W. Cao, and S. X. Yang, "An improved potential game theory based method for multi-UAV cooperative search," *IEEE Access*, vol. 8, pp. 47787–47796, 2020, doi: 10.1109/ACCESS.2020.2978853.

[44] P. Garcia-Aunon, J. del Cerro, and A. Barrientos, "Behavior-based control for an aerial robotic swarm in surveillance missions," *Sensors*, vol. 19, no. 20, p. 4584, Oct. 2019, doi: 10.3390/s19204584.

[45] K.-Y. Tsao, T. Girdler, and V. G. Vassilakis, "A survey of cyber security threats and solutions for UAV communications and flying ad-hoc networks," *Ad Hoc Netw.*, vol. 133, Aug. 2022, Art. no. 102894, doi: 10.1016/j.adhoc.2022.102894.

[46] J. Villagra, M. Clavijo, A. Díaz-Álvarez, and V. Trentin, "Motion prediction and risk assessment," in *Decision-Making Techniques for Autonomous Vehicles*. Amsterdam, The Netherlands: Elsevier, 2023, ch. 4, pp. 61–101, doi: 10.1016/B978-0-323-98339-6.00002-6.

[47] M. N. Huhns and L. M. Stephens, "Multiagent systems and societies of agents," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1999, pp. 79–120.

[48] A. C. Chapman, "Control of large distributed systems using games with pure strategy Nash equilibria," Ph.D. dissertation, Electron. Comput. Sci., Univ. Southampton, Londonm U.K., 2009. [Online]. Available: https://eprints.soton.ac.uk/69169/1/Thesis_Final.pdf

[49] D. Wolpert and K. Tumer, "An overview of collective intelligence," in *Handbook of Agent Technology*. Cambridge, MA, USA: MIT Press, 1999.

[50] D. Monderer and L. S. Shapley, "Potential games," *Games Econ. Behav.*, vol. 14, no. 1, pp. 124–143, May 1996, doi: 10.1006/game.1996.0044.

[51] G. Arslan, J. R. Marden, and J. S. Shamma, "Autonomous vehicle-target assignment: A game-theoretical formulation," *J. Dyn. Syst., Meas., Control*, vol. 129, no. 5, pp. 584–596, Sep. 2007, doi: 10.1115/1.2766722.

[52] A. C. Chapman, R. Anna Micillo, R. Kota, and N. R. Jennings, "Decentralized dynamic task allocation using overlapping potential games," *Comput. J.*, vol. 53, no. 9, pp. 1462–1477, Nov. 2010, doi: 10.1093/comjnl/bxq023.

[53] J. S. Shamma, "Learning in games," in *Encyclopedia of Systems and Control*. Cham, Switzerland: Springer, 2021, doi: 10.1007/978-3-030-44184-5_34.

[54] J. R. Marden and J. S. Shamma, "Game-theoretic learning in distributed control," in *Handbook of Dynamic Game Theory*. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-44374-4_9.

[55] J. R. Marden and J. S. Shamma, "Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation," in *Proc. 48th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Monticello, IL, USA, Sep. 2010, pp. 1171–1172, doi: 10.1109/ALLERTON.2010.5707044.

[56] A. Muralidharan, Y. Yan, and Y. Mostofi, "Binary log-linear learning with stochastic communication links," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Tampa, FL, USA, Oct. 2015, pp. 1348–1353, doi: 10.1109/MILCOM.2015.7357632.

[57] L. B. V. de Amorim, G. D. C. Cavalcanti, and R. M. O. Cruz, "The choice of scaling technique matters for classification performance," *Appl. Soft Comput.*, vol. 133, Jan. 2023, Art. no. 109924, doi: 10.1016/j.asoc.2022.109924.

[58] *Mesa Overview*. Accessed: May 30, 2024. [Online]. Available: https://mesa.readthedocs.io/en/stable/overview.html

[59] A. Pellegrini, P. D. Sanzo, B. Bevilacqua, G. Duca, D. Pascarella, R. Palumbo, J. J. Ramos, M. À. Piera, and G. Gigante, "Simulation-based evolutionary optimization of air traffic management," *IEEE Access*, vol. 8, pp. 161551–161570, 2020, doi: 10.1109/ACCESS.2020.3021192.

[60] P. Mertikopoulos, Y.-P. Hsieh, and V. Cevher, "A unified stochastic approximation framework for learning in games," *Math. Program.*, vol. 203, nos. 1–2, pp. 559–609, Jan. 2024, doi: 10.1007/s10107-023-02001-y.

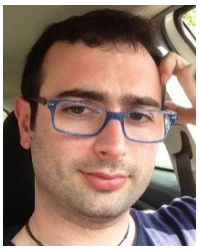
VITTORIO U. CASTRILLO was born in Caserta, Italy, in 1979. He received the M.S. degree (Hons.) in electronic engineering from the University of Naples Federico II, Italy, in 2003.

In 2004, he joined the Electronics and Communications Unit, Italian Aerospace Research Centre (CIRA). In the beginning, he was involved in the design and integration of data acquisition systems. Afterward, he worked on the design and integration of communication systems for unmanned aerial and space vehicles. Since 2011, his activities have been focused on the design of algorithms for base-band digital signal processing for communication systems with implementation on FPGA-based platforms. Recently, he has worked on the full (analog and digital) circuit design of complex electronic systems based on FPGAs, DDR memories, and microcontrollers. He is currently with the Security of Systems and Infrastructures Laboratory and manages a project for the development of innovative technologies for small drones.

DOMENICO PASCARELLA was born in San Felice a Cancello, Italy, in 1983. He received the B.S. and M.S. degrees in information engineering and the Ph.D. degree in electronics and information engineering from the Department of Industrial and Information Engineering, 'Seconda Università degli Studi di Napoli, Italy, in 2007 and 2016, respectively.

Since 2008, he has been a Researcher with the Italian Aerospace Research Centre (CIRA), Italy. From 2015 to 2020, he was with the Intelligent Systems Laboratory. Since 2020, he has been the Head of the System and Infrastructure Security Laboratory, CIRA. He has published scientific papers in journals and conference proceedings about the following themes: multiagent systems and automated planning for drones, security applications for drones, and modeling of complex systems. He was a part of the CIRA Team who awarded the 2020 Defence Innovation Prize assigned by the European Defence Agency.

GIANPAOLO PIGLIASCO was born in Naples, Italy, in 1973. He received the degree (Hons.) in computer engineering (specializing in automation) from the University of Naples Federico II, Italy, the first master's degree in enterprise innovation from the Italian Business School ''Fondazione CUOA,'' and the second master's degree in innovative methods for aeronautical maintenance from the Italian Aerospace Research Centre (CIRA), Capua. Since 2008, he has been a Researcher in the fields of applications for aeronautics and aerospace, participating in several important projects with regard to data analysis, process automation, design of intelligent systems for avionics, and telecommunications. Since 2020, he has also been a member of the System and Infrastructure Security Laboratory, CIRA.



IVAN IUDICE was born in Livorno, Italy, in November 1986. He received the B.S. and M.S. degrees in telecommunications engineering and the Ph.D. degree in information technology and electrical engineering from the University of Napoli Federico II, Italy, in 2008, 2010, and 2017, respectively.

He first served as a part of the Electronics and Communications Laboratory. Since November 2020, he has been with the Security of Systems and Infrastructures Laboratory, Italian Aerospace Research Centre (CIRA). He is involved in several international projects. He is the author of several papers in refereed journals and international conferences. His research interests include signal and array processing for communications, with current interests focused on physical-layer security, space-time techniques for cooperative communications systems, and reflective intelligent metasurfaces. He serves as a reviewer for several international journals and a TPC member for several international conferences.

ANGELA VOZELLA received the Ph.D. degree.

She is currently the Head of the Reliability Availability Maintainability Safety and Security (R.A.M.S.&S.) Department, Italian Aerospace Research Centre (CIRA). With 33 years of professional experience in system reliability engineering for safety-critical applications, she has been an active participant in many projects funded by the European Commission and the project manager of collaborative projects in space and transport. She was involved in many research initiatives, working groups, and committees supporting the definition and implementation of approaches to foster the fulfillment of Flightpath2050 goals with the main stakeholders in the aviation domain, such as EASA, ICAO, national civil aviation authority, military stakeholders, industry, and research entities. She is currently involved in many projects implementing innovative paradigms and related concepts in aerospace to face with safety, security, performance, and sustainability challenges, such as SESAR, Horizon Europe, Clean Aviation, EDF, and national-funded initiatives.