Multi-Agent Coordinated Close-in Jamming for Disabling a Rogue Drone

Panayiota Valianti®, Savvas Papaioannou®, Member, IEEE, Panayiotis Kolios®, Member, IEEE, and Georgios Ellinas®, Senior Member, IEEE

Abstract—Drones, including remotely piloted aircraft or unmanned aerial vehicles, have become extremely appealing over the recent years, with a multitude of applications and usages. However, they can potentially present major threats for security and public safety, especially when they fly across critical infrastructures and public spaces. This work investigates a novel counter-drone solution by proposing a multi-agent framework in which a team of pursuer drones cooperate in order to track and jam a rogue drone. Within the proposed framework, a joint mobility and power control solution is developed to optimize the respective decisions of each cooperating agent in order to best track and intercept the moving rogue drone. Both centralized and distributed variants of the joint optimization problem are developed and extensive simulations are conducted to evaluate the performance of the problem variants and to demonstrate the effectiveness of the proposed solution.

Index Terms—Multi-agent tracking, jamming, decision and control, optimization

1 INTRODUCTION

According to recent drone-related technology research [1], over the next few years the demand for consumer drones (i.e., unmanned automated vehicles (UAVs) and remotely piloted aircraft systems (RPASs)) will skyrocket. In particular, the expected demand for consumer drones will reach 7.8 million in 2020 with $3.3 billion in revenue, and this demand is expected to grow significantly over the next few years [2].

However, drones, like every new emerging technology, can potentially introduce new threads and risks for the safety of the public. Indeed, consumer drones are already perceived as a major threat especially when they fly in restricted airspaces, such as critical infrastructures (airports, ports, etc.) and public spaces (including shopping malls and stadiums). For instance, numerous times airports have shut down because of rogue drones, causing long delays to the flight schedules and significant inconvenience for airline passengers [3].

Unfortunately, no adequate solutions exist for the problem of rogue drones. To date, the relevant research community has focused on novel detection techniques [4] (including RF signal sniffing, computer vision, and sensor fusion), and interception techniques [5], including net-casting, RF denial systems, high-power lasers, and even trained eagles to capture the rogue drones [6]. As reported in [7], solutions for intercepting drones are still in their infancy and considerable amount of work still needs to be performed to ensure that drones do not pose any security or public safety threats. Indeed, interception technology is currently immature and drones equipped with reliable and accurate detection and tracking, as well as interception capabilities are readily required (mainly because the proximity of the interception reduces the safety risk for nearby devices) [8].

To highlight the problem at hand, just recently a malicious attempt was made to disrupt air traffic at London’s Heathrow airport, by flying drones in the airport’s restricted airspace [9]. The disruption was successfully countered by British police, which used jamming technology to prevent the drones from taking off. According to this report [10], jamming is the most common method for countering rogue drones that is being explored by police and military personnel across the world. However, the Federal Aviation Administration (FAA), the governmental body of the United States that regulates all aspects of civil aviation, voiced recently some concerns about using jammers near airports, since they may impact airport operations [11].

In accordance, this work proposes a multi-agent cooperative framework in which a team of autonomous drones cooperate in order to track and jam a rogue drone (target). More specifically, the focus is on a realistic scenario in which the rogue drone follows a trajectory which is unknown to the agents and which needs to be estimated using noisy sensor measurements. In addition, it is assumed that the mobile agents exhibit a limited sensing range and they can detect the presence of the rogue drone inside their sensing range based on a probability function. Due to multipath, electronic interference, and thermal noise, the assumption in this work is that in addition to the target measurements the agents receive false-alarm measurements (i.e., clutter), which makes the problem of tracking and disabling the rogue drone even more challenging.
It should be noted that the use of multiple drones to track-and-jam a single rogue drone (target) has several advantages in comparison with the existing counter-drone approaches that mainly rely on terrestrial solutions, which are relatively static. In contrast to static approaches, our dynamic approach: (i) provides improved target tracking performance, as the sensors are able to dynamically move to positions where the tracking performance is maximized; (ii) enables the flexibility in target jamming, due to the mobility of the jammers; and (iii) localizes the interference power of the jamming power which ensures the safety of the surrounding communication systems.

The main objectives of the agents are: (a) to cooperate in order to accurately track the rogue drone over a period of time, and (b) to disable the rogue drone in the air by cooperatively transmitting power from their on-board antennas to jam the communication and sensing receivers of the rogue drone, while at the same time ensuring that the received power between the cooperating agents is kept below interfering levels. The specific contributions of this work are the following:

- A novel multi-agent framework for the joint tracking and jamming problem is introduced that captures all important characteristics of the scenario including the agent limited sensing range and detection uncertainty, the existence of clutter, and the unknown target trajectory.
- A novel mathematical programming model is derived to formulate the joint tracking and jamming problem, and centralized, decentralized, and distributed algorithms are developed to solve this problem.
- Extensive simulation results demonstrate the applicability of the proposed framework and evaluate the proposed algorithms for a large set of parameters to thoroughly conduct a comparative evaluation of the performance of the different proposed algorithms.

The rest of the paper is organized as follows. Section 2 describes the related work, while background details of the required mathematical theory are provided in the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TMC.2021.3062225. Thereafter, Section 3 includes models that describe the system dynamics and elaborates on the important characteristics of the scenario including the agent limited sensing range and detection uncertainty.

Fig. 1. Related work on countering a rogue drone.

### 2 Related Work

Existing literature on the problem of countering a rogue drone near sensitive areas, like airports, can be found in [4], [8], [10], [12] and is summarized in Fig. 1.

Rogue drone detection and tracking techniques rely on radars, RF signals, computer vision, acoustic signals and sensor fusion [4]. Most current solutions are based on fixed terrestrial sensors; for example, [13] describes a solution that uses radars, while [14] fuses acoustic signals, computer vision, and RF signals. Unfortunately, these systems are subject to limitations, especially in urban areas, due to obstacles. Recently, the authors in [15] proposed a dynamic radar network (DRN), composed of radars mounted on drones, that is able to distributively detect and track rogue drones in real time. A tracking algorithm is also presented in [16] as an application of a surveillance-evasion game.

Several defense techniques have been proposed that aim to interdict a rogue drone, where a distinction has been made between electronic measures, which are soft-measures, e.g., jamming, and physical defenses which are hard measures, e.g., shooting down the drone. Additionally, some works combine several defense elements.

One approach to address the interdiction problem through physical defenses is to use interceptor drones. Pursuit-evasion problems can represent conflicts between adversarial drone teams, when an intelligent evader (rogue drone) has full knowledge of the environment and is aware of the pursuer’s (defender drone) location and intent. Recently, authors have investigated several geometric approaches to solve for pursuit-evasion problems which generally rely on Voronoi partitioning. More specifically, in [17], [18] the authors consider the problem of drone pursuit-evasion in a 2D world, in which a number of pursuers are attempting to capture a single evader. The proposed algorithms try to minimize the area of the evader and demonstrate that the capture of the evader can be guaranteed when at least one of the pursuers approaches the evader within a specific distance. Further, a more recent work in [19] investigates the same problem and extends it to multiple evaders. In [20], the authors look into the interception problem with the additional constraint that the pursuer
drones are not permitted to enter no-fly zones. When the evader enters a no-fly zone, the pursuers are positioned on the perimeter of the zone to quickly capture the evader once it emerges from the no-fly zone.

A more recent work in [21] obtains cooperative pursuit strategies for the multi-pursuer-single-evader problem, where the evader is captured when it comes within a certain distance from the nearest pursuer. Other works address the single-pursuer-multiple-evader problem while assuming exact capture (distance is zero), including [22] where a fast pursuer is attempting the successive capture of multiple evaders in minimal time. An extension of this work proposes a less computationally expensive approach for the same problem [23]. Further, in [24] the evaders cooperate to avoid a hidden pursuer and aim to maximize the overall capture time for the entire group.

Physical defense utilizing interceptor drones can also be formulated as a reach-avoid game [25], where efficient computational methods for the multiplayer case exist [26], [27], [28], [29]. Unlike pursuit-evasion problems, in reach-avoid problems a pursuer attempts to intercept an evader, by occupying the same point, before the evader reaches its goal [25].

The authors in [30] developed an approach where a swarm of pursuer drones are positioned in a formation around the rogue drone in such a way to restrict its movement and escort it outside the restricted area by exploiting the rogue drone’s collision avoidance system. In that work, the authors assume that a high-quality tracking system is in place that is able to accurately detect and track the rogue drone.

The works in [31] and [32] consider the problem of intercepting a target UAV’s trajectory. The algorithm developed in [31] generates an interception trajectory for quadrotors in real time, while in [32], a reactive maneuver policy for interception is learned through reinforcement learning (RL). In particular, in that work, an RL pursuer is trained against a greedy evader in 2D.

Rothe et al. [33] combine two physical defense elements for catching a rogue drone with a net carried by cooperative UAVs. However, a target detection system is assumed, which provides reliable position and velocity estimates.

In order to prevent rogue drones from entering restricted airspaces, some works in the literature also consider electronic defenses which try to take advantage of vulnerabilities in the drone subsystems in order to hack them [7]. In particular, the authors in [34] exploit WiFi-enabled drone vulnerabilities to launch network-based attacks on the drone. Furthermore, the authors in [35] attack the rogue-drone’s dynamic state estimation by exploiting the vulnerabilities of common state estimation algorithms in order to misguide its navigation and therefore prevent it from entering a restricted airspace. Further, the works in [13] and [14] present a fixed on-ground RF signal jamming approach that disrupts the communication link between the drone and its operator in order to cause a loss in control.

The work described in this paper complements the aforementioned research by investigating the use of a swarm of pursuer drones that aim to intercept one or more rogue drone(s) using signal jamming. The novelty of the proposed approach (compared to the existing literature) lies in the joint mobility and power control solution presented for accurately tracking the moving drone, while at the same time maximizing the power received at the target, so as to jam its communication and sensing circuitry and prevent it from completing its mission.

In the Appendix, available in the online supplemental material, we provide a brief overview on the theory of random finite sets (RFS) and stochastic Bayesian filtering that will be used in this work. A more detailed description of these concepts can be found in [36], [37], [38], [39].

3 PROBLEM DEFINITION

As exemplified in Section 1, this work proposes a joint mobility and power control solution to counteract the operation of rogue drones. The objective is to make optimized decisions on the mobility and power control actions of each cooperating agent in order to best track the moving drone target, while at the same time maximizing the received power at the target in order to jam its communication and sensing circuitry.

The system model assumed in this work illustrating the interaction between the different models utilized in this work to enable the track-and-jam functionality proposed in this work is shown in Fig. 2. More specifically, throughout this work the following models were used.

3.1 Target Dynamic State-Space Model

Initially, a state-space model is used to represent the dynamic system. The state-space model focuses on the state vector of the system, which contains all the relevant information required to describe the system under investigation. In this section, we consider the stochastic filtering problem in a state-space form, which consists of a transition equation, i.e., Eq. (1), and a measurement equation, i.e., Eq. (2), where \( x_t \) are the states and \( y_t \) are the measurements.

Specifically, in this work it is assumed that the target state vector \( x_t \in \mathbb{R}^5 \), that represents the actual kinematics of the target, is defined as \( x_t = [x, \dot{x}, y, \dot{y}, \omega]_t \), where \( (x, y) \) give the 2D position of the target in Cartesian coordinates, \( (\dot{x}, \dot{y}) \) are the velocities of the target in the \( x \) and \( y \) direction
respectively, and $\omega$ is the turn rate. Additionally, the target measurement vector, i.e., $y_t = [\phi, r]^T \in \mathbb{R}^2$ is composed of noisy heading and range observations. That said, we consider that the target state-space model is of the form[40]

$$x_t \sim p(x_t|x_{t-1})$$

$$y_t \sim p(y_t|x_t, s_t),$$

where $x_t \in \mathbb{R}^{n_x}$ is the target state at time $t$ (i.e., hidden random variable), $y_t \in \mathbb{R}^{n_y}$ is the target measurement, and $s_t \in \mathbb{R}^{n_s}$ is the state of the agent (defined in the next section). The target states follow a first order Markov process defined in terms of the transitional density $p(x_t|x_{t-1})$ and the dependence of the measurement $y_t$ on the target and the agent state is modeled by the conditional density $p(y_t|x_t, s_t)$, i.e., the likelihood function. It should be noted that both Eqs. (1) and (2) are assumed to follow a first order Markov process, so that the widely accepted Bayesian filtering equations, i.e., Eqs. (5) and (6) in the Appendix, available in the online supplemental material, can be used to compute the target estimation. More specifically, the target state evolves in time according to the following discrete time dynamics:

$$x_t = \zeta(x_{t-1}) + v_t,$$

where the non-linear function $\zeta: \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}$ models the dynamical behavior of the target and the process noise $v_t \in \mathbb{R}^{n_v}$ is normally distributed and models the random disturbances in the state evolution. The measurement equation is given by:

$$y_t = h(x_t, s_t) + w_t,$$

where the non-linear function $h: \mathbb{R}^{n_x} \times \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{n_y}$ defines the relationship between the states $(x_t, s_t)$ and the measurement $y_t$. The measurement noise $w_t$ is normally distributed and independent of $v_t$.

Finally, due to sensor imperfections, the agent also receives at each time $t$ a set $W_t = \{w^1_t, w^2_t, \ldots, w^n_t\}$, $w^l_t \in \mathbb{R}^2$ of false-alarm measurements, uniformly distributed inside the measurement space, whose size $n$ is distributed according to the Poisson distribution with parameter $\lambda$. That said, the agent receives at each time $t$ a collection of measurements (i.e., at most one target measurement plus a random number of false-alarm measurements) $Z_t = Y_t \cup W_t$ where $Y_t \subset \{0, y_t\}$. In Section 4, we show that $Z_t$ can be modeled as a random finite set (RFS) [39] and we derive the RFS measurement likelihood function $f_Z(Z_t|x_t, s_t)$ for the problem studied in this work.

### 3.2 Agent Dynamics

Suppose that we have at our disposal a set of controllable mobile agents $S = \{1, 2, \ldots, |S|\}$, where $|S|$ denotes the cardinality of the set, i.e., the number of available agents. Each agent $j \in S$ is subject to the following discrete time dynamics:

$$s^j_t = s^j_{t-1} + \left[ \begin{array}{c} l_1 \Delta_R \cos(l_2 \Delta_R) \\ l_1 \Delta_R \sin(l_2 \Delta_R) \end{array} \right], \quad l_1 = 0, \ldots, N_R, \quad l_2 = 0, \ldots, N_s,$$

where $s^j_{t-1} = [s^j_x, s^j_y]^{T-1} \in \mathbb{R}^2$ denotes the position of agent $j$ (i.e., $xy$-coordinates) at time $t - 1$, $\Delta_R$ is the radial step size, $\Delta_U = 2\pi/N_U$, and the parameters $(N_U, N_R)$ specify the number of possible control actions. We denote the set of all admissible control actions of agent $j$ at time $t$ as $U^j_t = \{s^j_1, s^j_2, \ldots, s^j_{|U^j|}\}$ as computed by Eq. (5). An illustrative example is shown in Fig. 3.

### 3.3 Agent Sensing Model

The agent exhibits a limited sensing range for detecting nearby targets which is modeled by the function $p_D(x_t, s_t)$. This function gives the probability that a target with state $x_t$ at time $t$ is detected by an agent with state $s_t = [s^j_x, s^j_y]^T \in \mathbb{R}^2$. More specifically, the target with state $x_t$ and 2D Cartesian coordinates $p_t = Hx_t$ (where $H$ is a matrix that extracts the $xy$-coordinates of a target from its state vector) is detected by an agent with state $s_t$ with probability which is given according to:

$$p_D(x_t, s_t) = \begin{cases} p_{D_{\text{Max}}} & \text{if } d_t < R_0 \\ \max\{0, p_{D_{\text{Max}}} - \eta(d_t - R_0)\} & \text{if } d_t \geq R_0, \end{cases}$$

where $d_t = \|Hx_t - s_t\|_2$ denotes the euclidean distance between the agent and the target, $p_{D_{\text{Max}}}$ is the detection probability for targets that reside within distance $R_0$ from the agent’s position, and parameter $\eta$ captures the reduced effectiveness of the agent to detect distant targets.

### 3.4 Path-Loss Model

Each agent carries an omnidirectional antenna which is the main mechanism used to jam the rogue drone. Since the drone-to-drone wireless aerial channels suffer from free space path-loss (FSPL) [4], [13], [41], the FSPL model with path-loss exponent 2 can be adopted, which is given as follows in dB:

$$A(d_t)[dB] = 20\log_{10}(d_t) + 20\log_{10}(f) + 32.45 - G_T - G_R,$$

where $d_t$ is the euclidean distance between the transmit and receive drones in meters at time-step $t$, $f$ is the carrier frequency, and $G_T$ and $G_R$ are the transmit and receive gains, respectively.
frequency in GHz, and $G_T$ and $G_R$ are the gains of the transmitter and receiver antennas in dB, respectively.

### 3.5 Joint Tracking and Jamming Objective

Let the function $\xi^j(x_t, u^j_t)$ denote the tracking cost of agent $j \in S$ with state $u^j_t$ when tracking a target with state $x_t$. Also, let $P^j\Lambda(x_t, u^j_t)$ be the path-loss experienced between agent $j$ and the target $x_t$, when the agent’s transmit power is $P^j$. Then, the joint mobility and power control decision problem for tracking and jamming the target can be described as follows:

$$
(P1) \max_{j \in S} \sum_{j \in S} \xi^j(x_t, u^j_t) + w_2 \sum_{j=1}^{\vert S \vert} P^j \Lambda(x_t, u^j_t) \tag{8}
$$

subject to

$$
\sum_{i \in S, i \neq j} P^i \Lambda(u^i_t, u^i_t) \leq \Delta^j, \forall j \in S \tag{9}
$$

$$
0 \leq P^j \leq P^j_{\text{max}}, \quad u^j_t \in \mathbb{U}^j_t, \quad 0 \leq P^j \leq P^j_{\text{max}}, \quad u^j_t \in \mathbb{U}^j_t, \tag{10}
$$

where the weighting parameter $w_i$, $i = 1, 2$ acts as a normalization factor between the two complementary objectives. Since omnidirectional transmissions are assumed in this work, the received power between cooperating agents must be maintained below some critical interference level, as indicated in Eq. (9), to ensure safe operation of the pursuers. The objective function in Eq. (8) enables the movement of the agents and their transmit powers to maximize tracking and jamming of the target, while the variable bounds in Eq. (10) ensure feasible solutions. It should be noted that for the optimal joint tracking and jamming problem, both tracking and jamming objective functions are necessary to derive the solution. One of the most commonly used classical approaches to solve a multi-objective optimization problem is the weighted sum method, which is precisely the method we used to define the objective function in Eq. (8). Specifically, it tries to find a balance between joint tracking and jamming a rogue drone by utilizing the weighted sum of the joint tracking cost (defined as the joint probability of detection in Section 4.2), and the jamming power received to the target.

As a motivating example for solving problem (P1), a simplified comparison study is presented below using the model definitions in Section 3 and considering a single target located at the center of a circle with radius $r$ and four agents placed around the target at: (a) equal distances between them on the circumference of the circle and (b) random distances (with the average distance to the target equal to $r$). Figs. 4 and 5 show the respective results for the aforementioned cases. Comparing the two figures, it is evident that for smaller $r$ the uneven distribution of agents enables higher received power at the target, while for higher values of $r$ the difference diminishes, mainly due to the non-linear effects of the path-loss model. However, since good quality tracking is achieved closer to the target, then the agents should aim to close-in on the target and thus arises the need to intelligently decide on their formation to ensure maximum received power. As a note, the average received interference to the cooperating agents is the same for both scenarios (as shown by the second plot in both figures) which is expected, since the average distance to the target is the same for both cases (i.e., $r$).

In the following section we explicitly define the track-and-jam functionality and provide algorithmic approaches to take multi-agent cooperative decision and control actions.

## 4 PROPOSED APPROACH

The proposed technique utilized in this work is outlined in Fig. 6. In summary, this approach decouples the joint mobility and power control decision problem (P1) and proposes a cascaded control architecture which consists of a tracking controller which gives input to a power controller. This cascaded controller is used at each time-step to determine the optimal mobility and power control actions for each agent. Each agent uses stochastic filtering to estimate the state of the rogue drone. The sections that follow below describe in detail all parts of the proposed system architecture to provide the track-and-jam functionality, as well as different algorithmic approaches that can be utilized allowing the agents to collaboratively decide about their mobility and power control actions.

### 4.1 RFS Estimation

As we have mentioned in Section 3.1, at each time step $t$ an agent $j$ with state $s_t$ may receive, in addition to the target measurement $y_t$, multiple false-alarm measurements (i.e., clutter) $W_t = \{w^j_1, w^j_2, \ldots, w^j_n\}$, where the number of elements in $W_t$ is random and varies over time. As a
Fig. 6. Proposed system architecture.

consequence, the measurements received by agent \(j\) can be modeled by the RFS \(Z_i\) as the union of two independent

\[
Z_i = Y_i(x_t, s_t) \cup W_t,
\]

where \(Y_i(x_t, s_t)\) is modeled as Bernoulli RFS which contains a single measurement from the target if the target is detected or is empty otherwise, and \(W_t\) is a Poisson RFS due to false-alarms. Subsequently, the target-generated RFS measurement likelihood function \(f_Y(Y_i|x_t, s_t)\) is given by:

\[
f_Y(Y_i|x_t, s_t) = \begin{cases} 1 - p_D(x_t, s_t) & \text{if } Y_i = \emptyset \\ p_D(x_t, s_t)p(y_t|x_t, s_t) & \text{if } Y_i = \{y_t\}. \end{cases}
\]

Thus, \(Y_i = \emptyset\) (i.e., no target measurement) with probability \(1 - p_D(x_t, s_t)\) and \(Y_i = y_t\) (i.e., the target measurement \(y_t\) has been received by the agent with state \(s_t\)) with probability \(p_D(x_t, s_t)p(y_t|x_t, s_t)\). On the other hand, the Poisson RFS \(W_t\) which models the set of false alarm measurements received by an agent at time \(t\) (with intensity function \(I(w_t) = \lambda c(w_t)\), where in this work \(c(.)\) denotes the uniform distribution over the measurement space) has a multi-object pdf which is given by:

\[
f_W(W_t) = e^{-\lambda} \prod_{w \in W_t} I(w) = e^{-\lambda |W_t|} \prod_{w \in W_t} c(w).
\]

In order to apply Bayes recursion (i.e., see Appendix, available in the online supplemental material) to estimate the conditional density \(p(x_t|Z_{1:t})\), where now \(Z_{1:t}\) denotes a sequence of RFS measurements through time, we need to derive the RFS likelihood function i.e., \(f_Z(Z_i|x_t, s_t)\).

To do that first observe that the RFS \(Z_i\) is the union of two independent RFSs i.e., \(Y_i\) and \(W_t\) and thus the joint RFS likelihood function \(f_Z(Z_i|x_t, s_t)\) can be computed by applying the general product rule (see [39] p.85) and taking into account the combinations of all mutually disjoint subsets of \(Z_i\). Thus for the two independent RFSs in this work

\[
f_Z(Z_i|x_t, s_t) = f_Y(\emptyset|x_t, s_t) \times f_W(Z_i) + \sum_{z \in Z_i} f_Y(\{z\}|x_t, s_t) \times f_W(Z_i \setminus \{z\}),
\]

where the operation \(Z \setminus \{z\}\) denotes the set subtraction. By substituting Eqs. (12) and (13) into Eq. (14), the expression \(f_Z(Z_i|x_t, s_t)\) can be obtained as

\[
f_Z(Z_i|x_t, s_t) = [1 - p_D(x_t, s_t)]e^{-\lambda |Z_i|} \prod_{z \in Z_i} c(z) + e^{-\lambda |Z_i| - 1} p_D(x_t, s_t) \sum_{z \in Z_i} p(z|x_t, s_t) \prod_{\bar{z} \not\in Z_i} c(\bar{z}).
\]

Observe from Eq. (15) that the first term accounts for the scenario where the agent receives no target-generated measurement and thus all the received measurements are due to false-alarms. On the other hand, the second term expresses the more general case where the agent receives one measurement from the target and multiple false alarm measurements. Eq. (15) can now be used as the measurement likelihood function in the Bayes recursion update step, to compute the posterior distribution of the target state.

It should be noted that in this work we use the Bayes recursion to (a) compute and propagate in time the posterior distribution of the target state in order to achieve target-tracking and (b) to extract the point estimate of the target state (from its distribution) and utilize it in our jamming control module to achieve uninterrupted radio jamming. Using the Bayes recursion to estimate the posterior filtering density of the target state is essential in this work since we assume (i) stochastic target dynamical model, (ii) noisy model for the target measurements, (iii) multiple false alarm measurements (whose number is random and changes over time) and finally (iv) a probabilistic target detection model. To account for the aforementioned uncertainties and to estimate in a rigorous way the probability density function of the target state we utilize in this work Recursive Bayesian Estimation, also known as the Bayes Filter which is described in this section and in the Appendix, available in the online supplemental material.
4.2 Cascaded Decision and Control Algorithm

The previous section described how to obtain the RFS likelihood function required in the update step of the Bayes filter in order to estimate the posterior distribution of the target state. This section describes how to actively control the movement of the agents (i.e., selecting their mobility control actions) in order to maintain tracking of the detected target. More specifically, we seek to find the optimal control actions \( u_j^t \in U_j^t, \forall j \in S \) that must be taken at time step \( t \) by each agent \( j \) so that the state of the target is estimated as accurately as possible.

First, observe that the RFS likelihood shown in Eq. (15) is conditioned on the state of the agent at time \( t \). In other words, the control action \( u_j^t \) taken by agent \( j \) at time \( t \) affects the received measurements \( Z_j^t \), which in turn affects the computation of the posterior distribution \( p(x_t|Z_j^t) \) in the update step of the Bayes filter. Thus, ideally, to optimally control the movement of the agents, we would require the knowledge of the future measurements \( Z_j^t \). In order to get around this problem we perform the following steps: First, note from Eq. (15) that agent \( j \) will receive a target measurement \( y_{j,t} \) only when the target resides within the agent’s sensing range and is detected with probability according to Eq. (6). As a result, choosing the control action which results in detecting the target and subsequently receiving a target measurement is a sufficient condition for maintaining tracking and estimating the target state. That said, let us denote the active control objective function for agent \( j \) as \( \xi^j(x_t, u_j^t) \), and so the optimization problem to solve now becomes:

\[
\begin{align*}
  u_j^t &= \text{arg max}_{u_j^t \in U_j^t} \xi^j(x_t, u_j^t) = \text{arg max}_{u_j^t \in U_j^t} p_D(x_t, u_j^t),
\end{align*}
\]

where \( x_t \) denotes the target state at time \( t \). We seek to find the control action \( u_j^t \) which results in maximizing the probability of observing the target with state \( x_t \). Since the target state \( x_t \) is also not available until the \( u_j^t \) is applied, we approximate it as

\[
  x_t \approx \tilde{x}_t = \text{arg max}_{x_t} p(x_t|Z_{1:t-1}).
\]

In essence, we use the predictive density \( p(x_t|Z_{1:t-1}) \) (i.e., Eq. (5) found in the Appendix, available in the online supplemental material, that calculates the predictive density of some hidden state \( x_t \) at time \( t \) given all measurements \( Z_{1:t-1} = Z_1 \ldots Z_{t-1} \) and we extract the most likely predicted target state for time \( t \). Then, we optimize Eq. (16) as if the target state is actually \( x_t = \tilde{x}_t \). Assuming that the agents are independent of each other, we define the active control as

\[
\begin{align*}
  u_1^t, \ldots, u_S^t &= \text{arg max}_{u_1^t, \ldots, u_S^t} \prod_{j \in S} \xi^j(\tilde{x}_t, u_j^t),
\end{align*}
\]

where \( u_j^t \in U_j^t \) and \( \tilde{x}_t \) is obtained from Eq. (17) for each agent via their predictive density \( p(\tilde{x}_t|Z_{1:t-1}) \) as explained in the previous paragraph.

The solution of the problem presented in Eq. (18) provides the optimal mobility controls for all agents with respect to target tracking. However, these controls are not necessarily the best for the objective of downing the rogue drone. More specifically, in order to capture and disable a rogue drone, the agents need to transmit a specific amount of power towards the target, while at the same time minimize the power interference between them. In order to achieve the joint objective of tracking and jamming the target, we utilize a cascaded control architecture (as illustrated in Fig. 6) in which we first find the optimal mobility control actions which result in a satisfactory tracking accuracy and then we perform a second optimization step where we refine further these mobility actions to achieve jamming control. To do that, instead of finding the mobility control actions which maximize Eq. (18), we first compute the set \( \Omega = \{\omega_1, \ldots, \omega_M\} \) of all combinations of actions \( u_j^t, \forall j \in S \), which satisfy

\[
\prod_{j \in S} \xi^j(\tilde{x}_t^j, u_j^t) > \vartheta,
\]

where \( \vartheta \in [0, 1] \) is the desired threshold of the joint probability of detection. In the second step, we select the optimal mobility control action and the level of transmit power for each of the cooperating agents to maximize the received jamming power at the target. This optimal control and decision problem is formulated according to (P2) below as an optimization problem

\[
\begin{align*}
  \text{(P2)max} \sum_{\omega \in \Omega} \sum_{j = 1}^|S| P_{\omega}^j \mathcal{A}(\tilde{x}_t^j, s_j^t) \\
  \text{s.t.} \sum_{i \in S, i \neq j} P_{\omega}^i \mathcal{A}(s_i^t, s_j^t) \leq \Delta^i, \forall j \in S, \omega \in \Omega \\
  b_{\omega}\mathcal{S} P_{\max} \geq \sum_{j} P_{\omega}^j \forall \omega \in \Omega \\
  \sum_{\omega} b_{\omega} = 1 \\
  0 \leq P_{\omega}^j \leq P_{\max}, \ b_{\omega} \in \{0, 1\},
\end{align*}
\]

(P2) is a mixed-integer non-linear problem where the best alternative mobility control action and the transmit power assigned for each agent are decided, in order to maximize the target received power, while ensuring that the interference level between cooperating agents is maintained below a threshold \( \Delta \), as described in constraint (21). Constraint (22) ensures that when a combination of control actions is selected, the total transmit power is curbed by the maximum transmit power \( P_{\max} \) of all agents. The selection of a single combination of control actions is ensured through Eq. (23). The decision variables of optimization problem (P2) are defined in Eq. (24).

Since (P2) is computationally hard to solve in practice, we devise hereafter a discretized instance of the problem that can be solved quickly for a small number of agents. In this case, instead of continuous power levels, we assume a predefined discrete set of power levels \( P_{\omega}^j \) where \( l \in \hat{P} \) is the available discrete power level from set \( \hat{P} \) for agent \( j \) in combination \( \omega \). Then, the problem is reduced to a combinatorial problem, where the best alternative combination \( \omega \) and the best alternative power level for each agent is chosen
for maximum received power at the target, while satisfying the interference levels between the cooperating agents. The resulting centralized discrete cascade control algorithm is presented below (Algorithm 1).

Algorithm 1. (Joint) Cascaded Decision and Control Algorithm

Input: $p_j(x_{t-1}|Z_{t-1}) \forall j \in S, P$
1: Compute the predictive density $p_j(x_t|Z_{t-1}) \forall j \in S$ using Eq. (5) from the Appendix, available in the online supplemental material.
2: Compute an estimate of the target state for time $t$ as $\hat{x}_t = \arg\max_{x_t} p_j(x_t|Z_{t-1}) \forall j \in S$ (i.e., Eq. (17)).
3: Define set $\Omega$ of all combinations of joint control actions, which satisfy the tracking accuracy constraint $\vartheta$ using Eq. (19).
4: Select $\omega \in \Omega$ for solutions that satisfy Eq. (21).
5: Compute an estimate of the target state for time $t$ as $\hat{x}_t = \arg\max_{x_t} p_j(x_t|Z_{t-1}) \forall j \in S$ (i.e., Eq. (17)).
6: Execute the optimal mobility control actions $u_{i_m}$ and transmit power $P_{i_m}$ \forall $j \in S$.
7: Receive the target measurement set $Z_t \forall j \in S$.
8: Compute posterior density $p_j(x_t|Z_{t-1}) \forall j \in S$ using Eq. (6) from the Appendix, available in the online supplemental material.
9: Estimate target state $\hat{x}_t$ as $\arg\max_{x_t} p_j(x_t|Z_{t-1}) \forall j \in S$.

Although the centralized Cascaded Decision and Control algorithm performs very well for a small number of agents, complexity is given by $O(|U||P|S)$ (i.e., $O(|U|C^2)$) in terms of the number of agents, $|U|$; number of mobility controls per agent, $|P|$; number of power levels per agent, i.e., it is exponential in terms of the number of agents. Therefore, the problem becomes intractable for a larger parameter space. The computation complexity arises from the fact that in this algorithm all combinations were evaluated and decisions were made and executed jointly. Since an exhaustive search of all discrete mobility and jamming states is carried out for all agents, Algorithm 1 ensures that the optimal combination of alternative actions is selected and executed. To reduce the complexity, we propose two approximation algorithms (i.e., sequential (Algorithm 2) and distributed (Algorithm 3) algorithms) that differ in the way the decisions are computed and executed. In particular, as discussed below, the sequential and distributed algorithms provide suboptimal solutions, while reducing Algorithm 1’s complexity to polynomial in terms of the number of agents, i.e., they process at most $O(|U||P|S)$ combinations (some mobility control actions $U$ are omitted after applying the detection threshold in Step 4 of both Algorithms 2 and 3).

4.3 Sequential Decision and Control Algorithm

Instead of computing the decisions jointly in a centralized manner, we propose a decentralized approach where at each time step the agents decide about their mobility and power control actions sequentially, i.e., one agent after the other makes decisions and then broadcasts its decisions to all other agents. Here, each agent $j$ that is currently deciding needs to compute the set $\Omega = \{\omega_1, \ldots, \omega_{|\Omega|}\}$ of its admissible control actions, which satisfy $\xi_j(\hat{x}_t, u_t^j) > \vartheta \ \forall u_t^j, j \in S$.

where $\omega_j = u_t^j \vartheta$ and $\vartheta \in [0, 1]$ is the desired threshold for the probability of detection.

Then, agent $j$ restricts its mobility control actions by allowing only the control actions that do not cause interference between a subset of the cooperative agents (as indicated in Eq. (21)) that have already made their control decisions (here we assume that all agents that already made a decision communicate both their control action and their transmit power level to all other agents). From the restricted set of control actions, agent $j$ greedily selects the mobility and power control actions that maximize the power received by the target. Finally, once all agents $\in S$ have made their control decisions, all actions are executed jointly. Algorithm 2 below details the steps involved in this procedure.

Algorithm 2. Sequential Decision and Control Algorithm

Input: $p_j(x_{t-1}|Z_{t-1}) \forall j \in S, P, \omega: \omega \leftarrow \emptyset, \vartheta \leftarrow 0$
1: Compute the predictive density $p_j(x_t|Z_{t-1}) \forall j \in S$ using Eq. (5) from the Appendix, available in the online supplemental material.
2: Compute an estimate of the target state for time $t$ as $\hat{x}_t = \arg\max_{x_t} p_j(x_t|Z_{t-1}) \forall j \in S$ (i.e., Eq. (17)).
3: for $j \in S$ in arbitrary sequence do
4: Define set $\Omega$ of $j$th agent’s admissible control actions $u_t^j \in U_j$, which satisfy $\xi_j(\hat{x}_t, u_t^j) > \vartheta$.
5: Define set $\Omega$ of solutions that satisfy Eq. (21) considering all preceding agent decisions.
6: Select $\omega' \in \Omega$ and $l \in P$ such that $\max_j \{P_{u_l^j} A(\hat{x}_t, s_l^j)\}$.
7: $\omega \leftarrow \omega \cup \{\omega'\}, l \leftarrow l \cup \{l\}$;
8: end for
9: Execute the optimal mobility control actions $u_t^m$ and transmit power $P_{i_m}$ \forall $j \in S$.
10: Receive the target measurement set $Z_t \forall j \in S$.
11: Compute posterior density $p_j(x_t|Z_{t-1}) \forall j \in S$ using Eq. (6) from the Appendix, available in the online supplemental material.
12: Estimate target state $\hat{x}_t$ as $\arg\max_{x_t} p_j(x_t|Z_{t-1}) \forall j \in S$.

It should be emphasized here that the simplicity of the sequential decision and control algorithm comes at a cost in performance. Note that since each agent makes decisions on its mobility and power control actions sequentially, the order in which agents make decisions greatly affects performance. Further, the subset of feasible control and transmit power decisions available to subsequent agents are affected by the actions taken by preceding agents. For example, when an arbitrary agent $i$ takes a decision on where to move and which power level to use (contributing in this way in maximizing jamming), the potential positions and power levels of subsequent agents are influenced by the interference caused by the signals transmitted by agent $i$. Hence, the simplicity of this greedy heuristic algorithm comes at the expenses of a sub-optimal solution that is greatly affected by the sequence of decisions being made. As a result, in some cases, solutions could be far from optimal (or even infeasible).

4.4 Distributed Decision and Control Algorithm

An alternative to sequential decision making is presented hereafter where agents distributedly decide on their control
and power levels based on the latest snapshot of the system state. Specifically, agent \( j \) requests, and retrieves, the state \( s_i | \forall i \in S, i \neq j \) and power control level \( P_{\text{ctrl}}^i | \forall i \in S, i \neq j \) from all other cooperating agents. Let snapshot \( S_j \), which is a tuple containing the state and power level of each agent, be defined as follows:

\[
S_j = \{ s_i | \forall i \in S, i \neq j \} \quad \text{and} \quad P_j = \{ P_{\text{ctrl}}^i | \forall i \in S, i \neq j \}.
\]

Using this snapshot, we define set \( \Omega \) of all control actions for agent \( j \) that do not cause interference between agent \( j \) and all other cooperating agents (as defined by Eq. (21)).

From the remaining control actions, the \( j \)th agent greedily selects the mobility and power control actions that maximize the power received by the target. Each agent executes its decisions once it computes them, therefore the execution is asynchronous. Algorithm 3 details the steps involved in this procedure.

**Algorithm 3. Distributed Decision and Control Algorithm**

**Input:** \( p^j(x_{t-1}|Z_{t-1}), \forall j \in S; \bar{P} \)

(Each agent \( j \in S \) executes the following steps)

1. Obtain a snapshot \( S_j \), defined in Eq. (25).
2. Compute predictive density \( p^j(x_t|Z_{t-1}) \) using Eq. (5) from the Appendix, available in the online supplemental material.
3. Compute an estimate of the target state for time \( t \) as \( \hat{x}_t = \arg \max_{x_t} p^j(x_t|Z_{t-1}) \) (i.e., Eq. (17)).
4. Define set \( \Omega \) of all admissible control actions \( u_t^i \in U_i^{\Omega} \), which satisfy \( \xi(\hat{x}_t, u_t^i) > \theta \).
5. Define set \( \check{\Omega} \subseteq \Omega \) for solutions that satisfy Eq. (21) considering the snapshot \( S_j \).
6. Select \( \omega \in \check{\Omega} \) and \( \ell \in \bar{P} \) such that \( \max_{\omega \in \check{\Omega}} \Lambda(\hat{x}_t, s_i) \).
7. Execute the optimal mobility control actions \( u_t^\omega \) and transmit power \( P_{\text{ctrl}}^\omega \).
8. Receive the target measurement set \( Z_t^i \).
9. Compute posterior density \( p^j(x_t|Z_t) \) using Eq. (6) from the Appendix, available in the online supplemental material.
10. Estimate target state \( \hat{x}_t \) as \( \arg \max_{x_t} p^j(x_t|Z_t) \).

Notably, under a practical setting, a plausible hybrid (centralized and distributed) system could be envisioned in which the system switches between the centralized (for higher performance) and distributed (for lower computational complexity) algorithms. This switch could be based on the instantaneous number of available agents and mobility and power level combinations, in order to obtain the maximum performance gains when also considering specific real-time constraints.

## 5 Evaluation

### 5.1 Simulation Setup

In order to evaluate the performance of the proposed approach, extensive simulation experiments were conducted to investigate the performance of the proposed solutions when varying the main parameters of the proposed framework (i.e., the number of agents, the mobility control, and the power levels).

In all simulations we assume that a single target maneuvers in an area of \( 100 \times 100 \) m and that the single target state at time \( t \) is described by \( x_t = [x, \dot{x}, y, \dot{y}, \omega]^\top \). for position, velocity, and angular turn rate components. For the target dynamics, i.e., Eq. (3), we assume that the target motion follows a coordinated turn (CT) model [42, 43] with variable angular turn rate and thus the target dynamics are given by \( x_t = \xi(x_{t-1}) + G \eta \) (note that for clarity in the results for our simulations we assumed that there is no process noise in the actual project trajectory). The function \( \xi(x) \) is given by

\[
\xi(x) = \begin{bmatrix}
    x + \frac{\sqrt{2}}{\sqrt{T}} \sin(\omega T) - \frac{\sqrt{2}}{\sqrt{T}} (1 - \cos(\omega T)) \\
    x \cos(\omega T) - \frac{\sqrt{2}}{\sqrt{T}} \sin(\omega T) \\
    y + \frac{\sqrt{2}}{\sqrt{T}} (1 - \cos(\omega T)) + \frac{\sqrt{2}}{\sqrt{T}} \sin(\omega T) \\
    x \sin(\omega T) + \frac{\sqrt{2}}{\sqrt{T}} \cos(\omega T) \\
    \omega
\end{bmatrix}
\]

while \( G \eta \) is given by

\[
G \eta = \begin{bmatrix}
    \frac{T^2}{2} & 0 & 0 \\
    T & 0 & 0 \\
    0 & \frac{T^2}{2} & 0 \\
    0 & T & 0 \\
    0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
    v_x \\
    v_y \\
    v_\omega
\end{bmatrix},
\]

where \( T = 1 \) s is the sampling interval, \( v_x \sim N(0, \sigma_v^2) \), \( v_y \sim N(0, \sigma_v^2) \), \( v_\omega \sim N(0, \sigma_\omega^2) \) with \( \sigma_v = 4 \) m/s and \( \sigma_\omega = \pi/180 \) rad/s. Once an agent detects a target it receives heading and range measurements, thus the measurement model of Eq. (4) is given by:

\[
h(x_t, s_t) = \left[ \text{arctan} \left( \frac{y - y_c}{x - x_c} \right), \| s_t - H x_t \|_2 \right] ,
\]

where

\[
H = \begin{bmatrix}
    1 & 0 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 & 0
\end{bmatrix} .
\]

The measurement likelihood function is then given by \( p(y_t | x_t, s_t) = N(y_t; h(x_t, s_t), \Sigma) \) and \( \Sigma \) is defined as \( \Sigma = \text{diag}(\sigma_x, \sigma_y, \sigma_\omega, \sigma_v, \sigma_v) \). The standard deviations \( (\sigma_x, \sigma_y, \sigma_\omega, \sigma_v, \sigma_v) \) are range dependent and given by:

\[
\sigma_x = \phi \beta \| s_t - H x_t \|_2, \\
\sigma_y = \phi \beta \| s_t - H x_t \|_2, \\
\sigma_\omega = \phi \beta \| s_t - H x_t \|_2,
\]

with \( \phi = \pi/180 \) rad, \( \beta = 10^{-5} \) rad/m, \( \gamma_0 = 1.5 \) m, and \( \beta_\omega = 3 \times 10^{-4} \) m\(^{-1}\). Moreover, the agent receives spurious measurements (i.e., clutter) with fixed Poisson rate \( \lambda = 10 \) uniformly distributed over the measurement space. The agent’s sensing model parameters take the following values: \( \beta_{\text{max}} = 0.99, \eta = 0.003, \text{and} \ R_0 = 10 \) m. The agent’s dynamical model has radial displacement \( \Delta_p = 3 \) m, \( N_R = 2 \), and \( N_\theta = 8 \) which gives a total of 17 control actions, including the initial position of the agent. The agents carry isotropic antennas \( G_T = G_R = 0 \) dB which at each time-step transmit jamming signals of discrete power levels from set \( \bar{P} = \{-30, -20, -10\} \) dBm, that have carrier frequency \( f = 2 \) GHz, or do not transmit any power. The joint probability of detection threshold \( \theta = 0.50 \) is applied over normalized values and the interference threshold is \( \Delta = -70 \) dBm. Finally, in order to handle the non-linear dynamics and measurement model, Bayes recursion in Eqs. (5) and (6) found in the Appendix, available in the online supplemental material, was implemented using
5.2 Simulation Results

We begin the evaluation of the proposed approach with a track-and-jam representative scenario with 3 agents and a single target, which takes place during 100 time-steps. The scenario is simulated once for each proposed algorithm (described in Sections 4.2, 4.3, and 4.4, for the joint, sequential, and distributed decision and control algorithms, respectively) and is depicted in Fig. 7. In this scenario, 3 agents and a target enter at \( k = 1 \) the 100 m \( \times \) 100 m area and track the target over time as shown in Figs. 7a, 7f, and 7k using each algorithm. The initial agent locations (marked with \( \Delta \)) are (60, 25), (55, 20) and (60, 15) for agents 1, 2, and 3 respectively, and the target initial state (marked with a circle) is (60, 0.9, 20, 0.2, 0.03).

At each time-step, the agents cooperate in order to track-and-jam the target. The agent maneuvers are shown in Figs. 7a, 7f, and 7k for the joint, sequential, and distributed algorithms, respectively. As shown in this set of figures, the sequential algorithm manages to maintain the shortest distance from the target. All algorithms provide satisfactory tracking performance as indicated by the optimal sub-pattern assignment (OSPA) error, which is shown in Figs. 7b, 7g, and 7l for each algorithm. Interestingly though, the distributed algorithm chose maximum power levels for each agent at almost each time-step (as shown in Fig. 7m), while the joint and sequential algorithms had more alterations (as indicated in Figs. 7c and 7h, respectively). Despite this, by utilizing the joint algorithm the target receives, at each time-step, more jamming power (Fig. 7e) than when the sequential and distributed algorithms are used (Figs. 7j and 7o).

This is reasonable, since the joint algorithm approximately performs an exhaustive search over the state-space of all combinations of mobility and power control actions between the agents at each time-step. In terms of agent interference, the joint and distributed algorithms seem to maintain equal received power from all other agents while for the sequential algorithm agents experience a higher variance in their received interference levels, as shown in Figs. 7d, 7i, and 7n. As expected, the joint algorithm manages to send out the highest power level to the target while the sequential algorithm greatly underperforms even compared to the distributed alternative as indicated in Figs. 7e, 7j, and 7o.

To investigate the average performance of the proposed architecture, we run 50 Monte Carlo simulations while varying the number of mobility control actions per agent, the number of agents, and the number of available power levels per agent. For all setups, we assume that the scenario lasts for 100 time-steps within which a target enters a 100 m \( \times \) 100 m area at \( k = 1 \).

With this setup, the first scenario considered the presence of three (3) agents with 9, 17, and 25 possible mobility control actions, while all other parameters remain the same as those defined in Section 5.1 (i.e., at each time-step either jamming signals are transmitted with power levels at \(-10\) or \(-20\) or \(-30\) dBm or jamming signals transmitted). Looking into the plots of Fig. 8, it is evident that a significantly different performance is observed between the three algorithms for all metrics, with the distributed algorithm approaching the performance of the joint algorithm, significantly outperforming its sequential variant. Further, it is also observed that the increasing number of controls only slightly affects the OSPA error for all three algorithms, with a greater impact on the sequential and distributed approaches. The same behavior is experienced with all other metrics as well, where an increasing number of controls greatly improves the performance of the sequential and the distributed algorithms, while the joint algorithm is less affected.

| TABLE 1  | Simulation Parameters                                                                 |
|----------|--------------------------------------------------------------------------------------|
| Target   | \( T \) Sampling interval 1 s                                                        |
| Dynamics | \( \sigma_x \) Standard deviation of process noise in dimension x 4 m/s\(^2\)        |
|          | \( \sigma_y \) Standard deviation of process noise in dimension y 4 m/s\(^2\)        |
|          | \( \sigma_\omega \) Standard deviation of process noise in angular turn rate \( \omega \) rad/s |
| Agent    | \( \phi_0 \) Parameters for standard \( \pi/180 \text{rad} \)                       |
| Measurement | \( \rho_0 \) deviation \( 10^{-3} \text{rad/s} \)                                  |
| Model    | \( \zeta_0 \) Parameters for standard 1.5 m                                        |
|          | \( \beta_0 \) deviation \( 3 \times 10^{-4} \text{m}^{-1} \)                         |
|          | \( \lambda \) Clutter receive rate 10                                               |
| Agent    | \( p_{\text{DM}}^{\text{max}} \) Maximum probability of detecting a target 0.99     |
| Sensing  | \( \eta \) Parameter for reduction in detection probability 0.003                    |
| Model    | \( R_0 \) Distance within which a target is detected with maximum probability \( p_{\text{DM}}^{\text{max}} \) 10 m |
|          | \( \theta \) Probability of detection threshold 0.50                                |
| Agent    | \( \Delta_R \) Radical step size 3                                                |
| Dynamics | \( N_R \) Parameters specifying the 2                                              |
|          | \( N_o \) number of possible control actions 8                                       |
| Wireless | \( \bar{P} \) Set of agent discrete power (-30,-20,-10) dBm or off                  |
| Jamming  | \( f \) Carrier frequency 2 GHz                                                    |
|          | \( G_T, G_R \) Transmitter and receiver gains 0 dB                                  |
|          | \( \Delta \) Interference threshold -70 dBm                                         |
Fig. 9 depicts the results of the second scenario, where the number of cooperative agents in the system increases from 2 to 3 and then to 4, while all other parameters remain the same as in Section 5.1 (i.e., 17 control actions and power levels at either $-30$ or $-20$ or $-10$ dBm and also including the case where no jamming signal is transmitted). As before, the joint algorithm is not considerably affected by the different configurations. The distributed algorithm has a similar
performance to the sequential algorithm as indicated by all metrics (i.e., OSPA error, transmit and received powers, agent interference), which improves as the number of agents decreases, due to the constraints posed by the interference levels among the agents. Nevertheless, the performance of the joint algorithm in terms of the target received power is significantly higher than that of the sequential algorithm, while the distributed algorithm approaches the performance of the joint algorithm (for the case of 2 agents).

The third scenario investigates the impact of the available transmit power levels on the performance of the pursuer drones. In this scenario, we consider 3 agents, 17 control actions, and power levels for the following cases: (i) Case 1: either a jamming signal at $-30$ dBm is sent or no jamming signal is sent, (ii) Case 2: either a jamming signal at $-30$ or $-20$ dBm is sent or no jamming signal is sent, and (iii) Case 3: either a jamming signal at $-30$ or $-20$ or $-10$ dBm is sent or no jamming signal is sent. As indicated in Fig. 10, all three algorithms experience similar performance for all metrics. More specifically, for the first two cases all algorithms chose the maximum transmit power level for almost each time-step, since the agent interference constraint is inactive, and thus achieved the same power received at the target. In the third case, where the agents need to intelligently decide upon their mobility and power controls in order to avoid interference between agents, the sequential algorithm underperforms in terms of target received power compared to the other two algorithms, while the distributed algorithm has a similar performance to the joint approach.
The aggregated performance of the three algorithms is summarized in Fig. 11, where the average target received power, agent interference, and execution time, which is on a logarithmic scale, for all simulation time-steps is shown. Evidently, the joint algorithm outperforms the other two solutions at the expense of significantly higher complexity and thus significantly higher execution time. Importantly though, the distributed algorithm provides comparable performance for a broad set of parameter values at significantly lower complexity (each agent makes its own decisions based on the instantaneous snapshot of the system state). The sequential algorithm provides the worst performance and could be improved by evaluating all possible sequences of agent decisions. However, the significant increase in computation and communication complexity cannot be justified when compared to the low complexity and high performance of the distributed approach. Thus, both sequential and distributed variants could be used in a real-world practical scenario due to their low execution time, whereas the joint variant serves as a benchmark, used mainly for comparison purposes.

Lastly, a simulation scenario that examines the performance of the distributed algorithm when an agent fails, either due to communication delays or a physical/electronic attack, is presented in Fig. 12. With the same setup as the first experiment of Section 5.2, a simulation is initially executed assuming no failures (results depicted in Figs. 12a, 12b, 12c, 12d, and 12e). Then, the same simulation is executed assuming that at time-step $k = 50$ the agent that is closest to the target, i.e., agent-1, fails. The results (shown in Figs. 12f, 12g, 12h, 12i, and 12j) demonstrate that despite losing one of the agents, the remaining agents quickly manage to bring back the jamming power delivered to the target at
pre-failure levels, and satisfactory jamming power is delivered to the target at each subsequent time-step.

6 Extension to Multiple Targets

In this section, we extend our proposed framework to demonstrate its applicability to the multi-target case. Due to space limitations we only present an extension of the distributed algorithm (Algorithm 3), however joint and sequential algorithms (Algorithms 1 and 2 respectively), can also be adjusted to the multi-target case.

Assume that there is a set of independent targets inside the area of interest $I = \{1, 2, \ldots, |I|\}$, the number of targets $|I|$ is fixed and known, and that the assignment of measurements to targets is also known. Basically, each agent $j$ maintains a separate density $p_j^i(\cdot)$ for each target $i$ at each time-step and the agents exchange their estimates and fuse them in order to obtain the final target state.

Let the function $\xi_j(x_i, u_j)$ denote the tracking cost of agent $j \in S$ with state $u_j$ when tracking target $i \in I$ with state $x_i$. Also, let $P_j^i(x_i, u_j)$ be the path-loss experienced between agent $j$ and the target $i$, when the agent’s transmit power is $P_j$.

Then, the joint mobility and power control decision problem for tracking and jamming multiple targets becomes:

\[
(P3) \max_{w_1} \prod_{i \in I} \prod_{j \in S} \xi_j(x_i, u_j) + w_2 \sum_{i=1}^{|I|} \sum_{j=1}^{|S|} P_j^i(x_i, u_j) \\
\text{s.t. } (9), (10),
\]

Fig. 10. Average performance results obtained for varying number of power levels (i.e., Case 1: either a jamming signal at $-30$ dBm is sent or no jamming signal is sent, Case 2: either a jamming signal at $-30$ or $-20$ dBm is sent or no jamming signal is sent, and Case 3: either a jamming signal at $-30$ or $-20$ or $-10$ dBm is sent or no jamming signal is sent) for the (i) Cascaded Decision and Control algorithm (10a - 10d), (ii) Sequential Decision and Control algorithm (10e - 10h), and (iii) Distributed Decision and Control algorithm (10i - 10l).
where $w_i, i = 1, 2$ are the weighting parameters. The objective function in Eq. (26) enables the movement of the agents and their transmit powers to maximize tracking and jamming of the targets, while the agent interference constraints in Eq. (9) ensure the safe operation of the agents and the variable bounds in Eq. (10) ensure feasible solutions (as described in Section 3.5 for the single target case). Clearly, more intelligent strategies can also be realized (e.g., maximize the minimum received jamming power, etc.); nevertheless, in this work we present an initial approach to the multi-target problem (including preliminary results) in order to demonstrate that the proposed framework can be extended to multiple targets.

The distributed algorithm has been modified to solve problem (P3) in an iterative way, and subsequently provide the multi-agent cooperative decision and control actions for countering multiple rogue drones. Hence, in steps 2, 3, 9, 10 of Algorithm 3, agent $j$ uses a separate density $p_j^i(.)$ and separate target estimations $\tilde{x}_i^j$ and $\tilde{x}_i^j$ for each target $i \in I$, instead of $p_j^i(.)$, $\tilde{x}_i^j$, and $\tilde{x}_i^j$. Also, in step 4 of Algorithm 3, a threshold of probability of detecting all targets $\prod_{i \in I} \xi'(\tilde{x}_i^j, u_i^j) > \theta$ replaces $\xi'(\tilde{x}_i^j, u_i^j) > \theta$, and in step 6 of Algorithm 3, an objective of maximizing the total power received at all targets $\max \sum_{t=1}^{[T]} P_{o,tl} \Lambda(\tilde{x}_i^j, s_i^j)$ replaces $\max P_{o,t} \Lambda(\tilde{x}_i^j, s_i^j)$. After step 10 of Algorithm 3, agent $j$ exchanges the target estimated states and covariances with all other cooperative agents and fuses them through covariance intersection [45].

Using the setup described in Section 5.1, we evaluate the proposed multi-target track-and-jam approach for the multi-target distributed algorithm, in a representative scenario with 3 agents and 2 targets, which takes place in a $100 \times 100$ m area for 100 time-steps. Fig. 13a presents the agent and target paths, while Fig. 13b indicates the tracking performance of individual and fused estimates (through the OSPA error), demonstrating that the agents manage to effectively track both targets (with a tracking error of less than 2 meters on average). Further, as observed in Figs. 13c and 13d, the agents manage to successfully jam both targets, while also maintaining their interference levels below the set threshold.

7 CONCLUSION

This work proposes utilizing a swarm of pursuer drones to track-and-jam one or more rogue drones. It proposes a cooperative framework where the pursuer drones jointly make optimized decisions on the mobility and power
control actions of each individual agent, while ensuring that the received power at the target is maximized. In this way, the rogue drone sensing and communication components can be efficiently jammed. Extensive performance evaluation results demonstrate that the problem can be efficiently solved using the developed distributed algorithm that provides good performance, albeit with much less complexity, compared to the joint solution, and significantly better performance compared to the sequential decision making approach. Future research avenues include incorporating secure coordination approaches [46], [47], [48] in our work, in order to preserve the communication between the agents from attacks and delays, and hence ensure their safe (without interference) and efficient (measurement exchange and fusion) operation. In addition, further investigation is needed to obtain more intelligent combinations of mobility and power controls, so as to devise more effective track-and-jam strategies for the multi-target scenario.

ACKNOWLEDGMENTS

This work was supported in part by the European Union Civil Protection under Grant agreement No 101017258 (SESAME). This work was also supported in part by the European Union’s Horizon 2020 research and innovation programme under Grant 739551 (KIOS CoE) and from the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

REFERENCES

[1] Goldman Sachs Research, “Drones: Reporting for work,” 2019. [Online]. Available: https://www.goldmansachs.com/insights/technology-driving-innovation/drones/
[2] Federal Aviation Administration, “FAA aerospace forecast: Fiscal years 2019–2039,” 2019. [Online]. Available: https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FY2019-39_FAA_Aerospace_Forecast.pdf
[3] D. Schneider, “Regulators seek ways to down rogue drones: Growing antidrone industry offers radar, remote ID, and other tools,” IEEE Spectrum, vol. 56, no. 4, pp. 10–11, Apr. 2019.
[4] I. Guvenc, F. Kooehler, S. Singh, M. L. Sichitiu, and D. Matolak, “Detection, tracking, and interdiction for amateur drones,” IEEE Commun. Mag., vol. 56, no. 4, pp. 75–81, Apr. 2018.
[5] J. Loeb, “Exclusive: Anti-drone technology to be tested in UK amid terror fears,” Eng. Technol., vol. 12, no. 3, pp. 9–9, Apr. 2017.
[6] BBC, “Eagles trained to take down drones,” 2016. [Online]. Available: https://www.bbc.com/news/av/world-europe-35750816/eagles-trained-to-take-down-drones
[7] K. Wesson and T. Humphreys, “Hacking drones,” Sci. Amer., vol. 309, no. 5, pp. 54–59, 2013.
[8] T. Humphreys, “Statement on the security threat posed by unmanned aerial systems and possible countermeasures,” Oversight and Management Efficiency Subcommittee, Homeland Security Committee, Washington, DC, US House, 2015.
[9] J. Johnson, “Heathrow protest thwarted as police use radiowaves to jam extinction rebellion group drones,” 2019. [Online]. Available: https://www.telegraph.co.uk/news/2019/09/13/two-arrested-within-grounds-heathrow-extinction-rebellion-group/
Savvas Papaioannou (Member, IEEE) received the BS degree in electronic and computer engineering from the Technical University of Crete, Chania, Greece, in 2011, the MS degree in electrical engineering from Yale University, New Haven, Connecticut, in 2013, and the PhD degree in computer science from the University of Oxford, Oxford, U.K., in 2017. He is currently a research associate at the KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia, Cyprus. His research interests include multiagent and autonomous systems, state estimation and control, multitarget tracking, probabilistic inference, Bayesian reasoning, and intelligent unmanned-aircraft vehicle (UAV) systems and applications. He is a member of the ACM and also a reviewer for various journals within the IEEE and ACM Associations.

Panayiotis Kolios (Member, IEEE) received the BEng and PhD degrees in telecommunications engineering from King’s College London, London, U.K., in 2008 and 2011, respectively. He is currently a research assistant professor at the KIOS Research and Innovation Center of Excellence, University of Cyprus. His research interests include both basic and applied research on networked intelligent systems. Some examples of such systems include intelligent transportation systems, autonomous unmanned aerial systems, and the plethora of cyber-physical systems that arise within the Internet of Things. Particular emphasis is given to emergency response aspects in which faults and attacks could cause disruptions that need to be effectively handled. He is an active member of the IEEE, contributing to a number of technical and professional activities within the Association.

Georgios Ellinas (Senior Member, IEEE) received the BS, MSc, MPhil, and PhD degrees in electrical engineering from Columbia University, New York. He is currently a professor and the past chair (2014-2020) of the Department of Electrical and Computer Engineering and a founding member of the KIOS Research and Innovation Center of Excellence, University of Cyprus. Previously, he served as associate professor of electrical engineering with the City College of New York (2002-2005), senior network architect with Tellium Inc. (2000-2002), and research scientist/senior researcher with Bell Communications Research (Bellcore) (1993-2000). He is a fellow of the IET (2019), a senior member of the OSA and ACM, and a member of the Marie Curie Fellows Association (MCFA). He has co-authored/co-edited four books on optical networks, more than 255 archived articles, conference papers, and book chapters, and he is the holder of 30 patents on optical networking. His research interests include optical/telecommunication networks, transportation networks, IoT, and critical infrastructure systems.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.