Multi-Robot Nonlinear Model Predictive Control for Persistent Monitoring

Francesca Pagano , Salvatore Marcellini, Mario Selvaggio, Vincenzo Lippiello , Fabio Ruggiero

***Abstract*—We present an approach to address a multi-robot persistent monitoring problem, where a team of agents must repeatedly survey specific points of interest (POIs) within a 2-D area. Our approach models the interest value of each POI with a heat-like dynamics. Each agent then solves online a nonlinear model predictive control (NMPC) problem to determine feasible trajectories that minimize the cumulative heat across all POIs. The trajectories are parameterized with Bézier curves, whose control points are used as optimization variables; this parametrization enables agents to efficiently communicate their optimized motions. An additional quadratic optimization layer adds safety guarantees while a central unit updates the global POIs' map. The method has been validated in simulation and real experiments, demonstrating that the algorithm can run online and on computationally limited hardware platforms. In addition, an extensive simulation campaign compares our NMPC against a state-of-the-art baseline across 90 randomly generated scenarios with different numbers of POIs. Our NMPC outperforms the baseline along the considered metrics, attaining lower robot velocities.**

***Index Terms*—Multi-robot, monitoring, coverage, Nonlinear Model Predictive Control.**

I. INTRODUCTION

MULTI-ROBOT systems promise increased efficiency in applications such as environmental monitoring [1], inspection of industrial plants, precision agriculture [2], and surveillance [3]. In this context, heterogeneous robotic teams have gained significant attention [4]–[8], as the deployment of robots with different capabilities, such as sensor types, mobility constraints, and endurance, can address complex and large-scale tasks with enhanced resilience and efficacy.

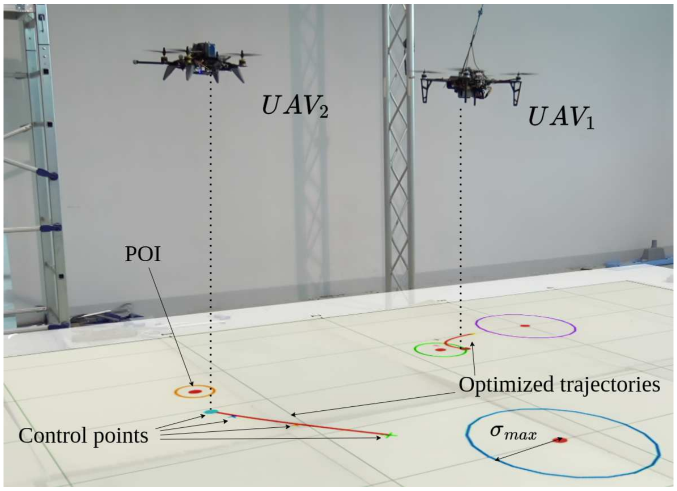


Fig. 1. Picture of the envisioned scenario. Two UAVs monitor repeatedly some POIs. Optimal trajectories and control points are depicted projected on the ground.

A substantial body of research addresses challenges in coverage and area search due to their significance in practical applications. While these tasks differ slightly, they both involve planning motions based on a distribution of information across the terrain [9], which may reflect physical quantities or the likelihood of a target's presence. Indeed, performing full coverage of an environment, unless strictly required by the task, is highly inefficient and time-consuming when some knowledge can be utilized to prioritize certain areas or to guide motions more effectively. In particular, spatial area functions are used in various domains: intruder detection and search-and-rescue missions [10] to represent the probability of locating or intercepting [11] a target; agricultural spraying to encode chemical release patterns [12]; cleaning robots to model dirt accumulation [13]; and environmental monitoring to indicate pollution levels, debris [14] or hazardous substance concentrations [15].

Despite significant advancements, deploying multi-robot systems in real-world scenarios remains challenging. Robots must adapt dynamically to changing conditions, optimize motion within safety and operational constraints, and use decentralized control for scalability and resilience. Furthermore, control frameworks should be flexible enough to accommodate heterogeneous robotic systems.

Within this context, this paper addresses a multi-robot persistent monitoring problem where a team of agents, possibly heterogeneous, has to survey some points of interest (POIs) repeatedly (see Fig. 1). These POIs might correspond to localized spots that must be observed, or the result of a

The research leading to these results has been supported by the COWBOT project, in the frame of the PRIN 2020 research program, grant number 2020NH7EAZ_002; the Agritech National Research Center and received funding from the European Union Next-Generation EU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1032 17/06/2022, CN00000022); and the AI-DROW project, in the frame of the PRIN 2022 research program, grant number 2022BYSBYX, funded by the European Union Next-Generation EU. (*Corresponding author*: Francesca Pagano)

Francesca Pagano is with Centro Servizi Metrologici e Tecnologici Avanzati (CeSMA), Corso Nicolangelo Protopisani, 80146, Naples, Italy (e-mail: francesca.pagano@unina.it).

Salvatore Marcellini is with Leonardo Innovation Labs, Leonardo S.p.A., 00195 Rome, Italy (e-mail: salvatore.marcellini@gmail.com). The research presented herein was conducted while the author was enrolled at PRISMA Lab. Leonardo S.p.A. was not involved in the development of this article or its content.

Mario Selvaggio, Vincenzo Lippiello, and Fabio Ruggiero are with PRISMA Lab, Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio 21, Naples, 80125, Italy (e-mail: mario.selvaggio@unina.it, vincenzo.lippiello@unina.it, fabio.ruggiero@unina.it).

non-uniform discretization of areas we want the robots to traverse [16]–[19]. Whenever the number of the POIs is less than or equal to the number of agents, the problem can be regarded as a task allocation or a static coverage. More interesting is the case in which the number of POIs exceeds the number of agents; thus, a static deployment [20] would be inefficient. The agents should then plan motions to maximize coverage over time, i.e., cover the highest number of POIs and minimize the time between the visits. In addition, the time spent on each POI may be affected by the time elapsed since the last visit. In this sense, the presented application shares some scopes with surveillance [21] and pursuit-evasion problems [22].

Based upon our previous work [23], our approach characterizes the POIs by their coordinates and a scalar value representing application-specific quantities. These values can correspond to the probability of detecting an intruder, a missing person, a gas leak, or model and information gain linked to a time-varying phenomenon to be monitored. The single POI's value function is shaped as a 2-D Gaussian that grows and expands while the agents are far from it; thus, the entire information distribution can be seen as a *heatmap* that dynamically varies depending on the current robots' locations. By making these quantities increase over time if no agent is near a point of interest, *persistent* monitoring is achieved, as shown later.

The use of a dynamically evolving quantity to characterize coverage quality is not new in the state-of-the-art; however, most approaches do not leverage an optimization horizon and often ignore the constraints inherent to real robotics systems. In contrast, we solve the monitoring problem using a nonlinear model predictive control (NMPC) that predicts the POI model's evolution and computes optimized, feasible motions to accomplish the task. A convenient motion parameterization allows the *online* solution of the nonlinear problem and additionally facilitates multi-robot coordination, even in heterogeneous settings. The proposed framework introduces a fundamentally different approach compared to current state-of-the-art methods, as detailed below.

A. Related works

Coverage problems usually leverage Voronoi partitioning of the environment, assigning to each robot a region of dominance [24]. Agents may be tasked with covering the whole region of interest either uniformly or with a given spatial density [25], [26], converging to an optimal deployment configuration, which may vary if the density changes in time [27] or if it is linked to a dynamical function [6]. In several works, the robots are then led to the center of their Voronoi partitions using controllers synthesized through gradient-based methods [25], model predictive control [28], and constrained quadratic problems [25]–[27].

However, coverage often assumes that the robots can cover the whole region, i.e., the union of their sensing areas can cover the entire domain. When such an assumption does not hold, more complex motion planning is needed, and the problem becomes closely related to coverage path planning, where an optimal sequence of waypoints must be computed [17], [18]. In these approaches, the visit order of the POIs is often determined offline using adaptations of the traveling salesman problem (TSP), which is NP-hard. For example, the approach in [18] solves a TSP to determine the POIs' visit order, performs a greedy optimization to compute the points observation periods, and then generates B-spline trajectories to ensure smooth and feasible motion for a single drone.

A variant of the coverage is also the so-called *awareness* coverage control, which introduces a *dynamic* function to represent robots' awareness of the domain [29]. This concept is explored in [30], and with an awareness loss term in [19], [31]–[34], where the awareness decays in time with a given constant, and the agents must move persistently over the domain. In particular, in [33], the awareness function is used as a time-varying density in a Voronoi-based coverage, and a local gradient-based control law is derived; a discrete cooperative controller based on the concept of cellular automata is proposed in [32]; while the work in [31] formulates a linear optimization problem to derive a periodic speed controller, still, the approach is not dynamic and only the leader agent moves along the path. In [19], periodic motions are computed optimizing agents' speeds and initial locations on a *given* closed-path connecting mission points. However, making robots move sequentially along a single closed path strongly restricts agents' mobility. Finally, the approach in [34] considers an unknown loss of awareness and proposes a combination of a gradient-based nominal controller and a perturbation one to balance exploitation and exploration. Similarly, a multi-robot search and rescue problem is addressed in [16] by defining a dynamic reward function based on Gaussian radial basis functions, which is used to generate gradient-based velocity references. None of the above approaches considers an optimization horizon nor the use of heterogeneous agents with different mobility constraints, as the presented work.

A combination of reward functions and variable time-step MPC is proposed in [11] for intercepting multiple targets, but only with a single UAV. In [9], an online receding horizon ergodic control approach is proposed for coverage, search, and target localization. Specifically, the approach leverages a continuous spatial distribution to represent information. This distribution is nonparametric and can be constant in time and known, or estimated by the agents. The methodology then employs local control computation and globally shared information so that the agents' trajectory statistics match the distribution ones.

B. Contribution

We propose a dynamic online motion planning approach based on NMPC for persistent monitoring of discrete points. Unlike most cited works, we propose a more complex control framework to account for constraints. Differently from [19], we do not employ closed paths, and we provide a solution that also allows static deployment cases. Our approach differs from [9] as we only consider localized spots of the domain while relying on a dynamical model to quantify and predict coverage cost evolution. The methodological contributions of our work are summarized in the following.

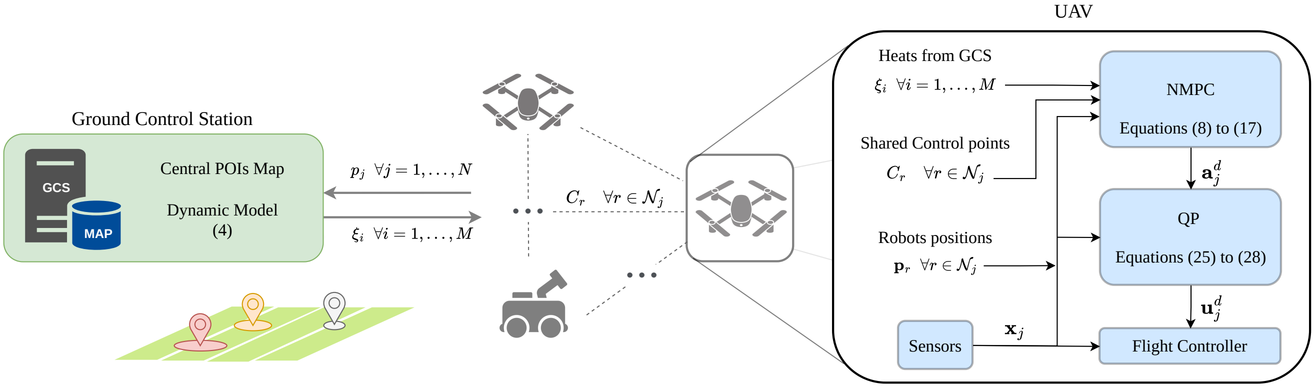


Fig. 2. System architecture—A ground control station (left) updates the global Points of Interest (POIs) values using agents' positions. The agents (center) retrieve the global map heat values $\boldsymbol{\xi}$ from the central unit and communicate their optimal trajectories through control points on a common network. Trajectories are obtained by locally solving a NMPC and a QP (right); the blue boxes denote local computation units, and use as inputs the global heats from the GCS, the shared control points, the robots' positions (which can either be globally shared or locally sensed), and on-board sensors measurements.

- We present an NMPC to perform persistent POIs monitoring, which generates locally optimal, parametrized motions for task execution. Building on our previous work [23], tailored for a single UAV, we propose a novel formulation that incorporates motion parameterization to enable efficient computation and broad applicability across different robotic systems.
- We generalize the approach presented in [23] to enable coordinated persistent monitoring with a multi-robot system. To this end, we: i) modify the POIs heat-like dynamics, making it dependent on multiple robots' positions; ii) propose a mixed centralized-decentralized framework in which all optimization problems are solved locally by each agent.

The overall framework is tested in realistic simulations across different case studies, with both homogeneous and heterogeneous robotic teams. A statistical analysis of the results obtained over 90 simulations demonstrates that our approach outperforms the baseline within different numbers of POIs in the single-robot scenario. Additionally, it is experimentally validated with two aerial robots.

II. PROBLEM STATEMENT

Consider a convex area $A \subset \mathbb{R}^3$, for simplicity modeled as a cube of dimensions $l_x \times l_y \times l_z$, that has to be surveyed by a team of $N > 0$ agents. The area contains $M > 0$ points of interest identified by their coordinates $\boldsymbol{\mu}_i = [\mu_{x_i}, \mu_{y_i}]^{\mathrm{T}} \in \mathbb{R}^2$ with $i = 1, \ldots, M$. When $M = N$, the optimal solution is trivial, as each agent should cover precisely one point; instead, for $M > N$, our objective is to cover all the POIs periodically. We assume that the POI distribution has been specified by the user or a higher-level task planning module, depending on the task, the current environment, and the operative conditions. We associate a continuous interest value to each point, whose dynamic depends on the agents' positions. From now on, such a value will also be referred to as *heat* due to the similarity of the crafted dynamic equation to the heat diffusion model, as will be detailed in Sec. III-D.

The control objective is to reactively control the agents' motions to minimize the heat values without violating the area bounds, the dynamic constraints, and avoiding collisions.

III. METHODOLOGY

This Section details the proposed methodology. Section III-A presents the overall architecture, the trajectory parameterization is introduced in Sec. III-B, while Sec. III-D presents the dynamic function associated with the points of interest. The discrete-time optimization problem is formulated in Sec. III-E and III-F, while the safety layer is presented in Sec. III-F.

A. System architecture

Figure 2 illustrates the system architecture. We consider a team of heterogeneous agents that can localize themselves within a common reference frame and communicate over a fully connected network. The system assumes the presence of a ground control station (GCS), serving as a central unit, that updates the POIs' values based on the agents' positions $\mathbf{p}_r \in \mathbb{R}^3, \;\; r = 1, \ldots, N$. All agents use the latest available values retrieved from the GCS, and the information shared by other agents to solve the NMPC locally, as better explained in the following sections.

B. Reference Trajectory

Differently from [23], motion trajectories are parameterized using cubic Bézier curves whose control points are used as decision variables in the optimization problem outlined in Sec. III-E. Given the optimized control points, the setpoints $\mathbf{p}^d(t) \in \mathbb{R}^3$ can be computed at each time instant by sampling

$$\mathbf{p}^d(t) = \sum_{p=0}^{3} \binom{3}{p} \left(1 - \frac{t}{T_f}\right)^{3-p} \left(\frac{t}{T_f}\right)^{p} \mathbf{c}_p, \qquad (1)$$

and its derivatives $\dot{\mathbf{p}}^d(t)$, $\ddot{\mathbf{p}}^d(t)$, where $t \in \mathbb{R}$ is the time variable, $\mathbf{c}_p \in \mathbb{R}^3$, for $p = 0, \ldots, 3$, are the four control points that describe the curve, $\binom{3}{p} = \frac{3}{p!(3-p)!}$ is the binomial coefficient, and $T_f > 0$ is the prediction horizon. In the

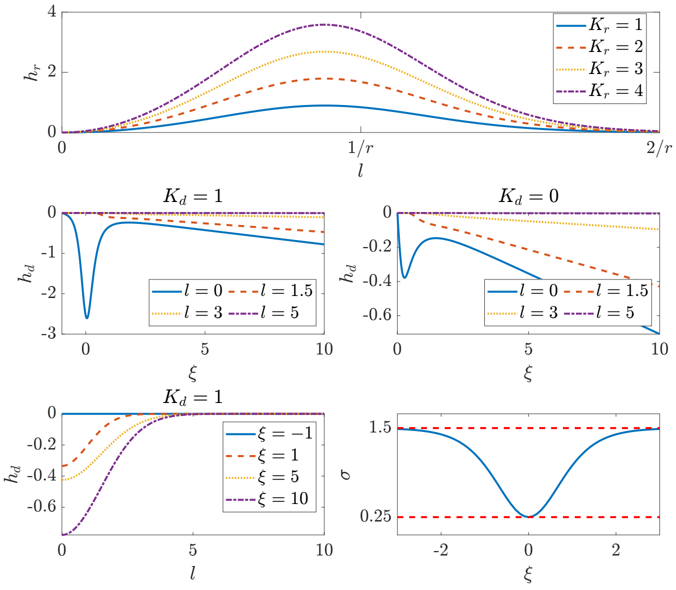


Fig. 3. POIs dynamics functions evolutions obtained with $l_{max} = 10$, $K_\sigma = 1$, $\sigma_{min} = 0.25$, $\sigma_{max} = 1.5$.

following, we will refer with $\mathbf{C} = \left[\mathbf{c}_0^{\mathrm{T}}, \mathbf{c}_1^{\mathrm{T}}, \mathbf{c}_2^{\mathrm{T}}, \mathbf{c}_3^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{12}$ to the stacked control points vector.

The trajectory parameterization reduces the optimization problem's dimensionality and introduces a continuously defined motion curve that can then be sampled at any desired frequency. Additionally, control points optimize intra-agent communication, as the full optimal reference trajectory can be communicated through $\mathbf{C}$.

C. Agent model

We assume that each robot is equipped with a low-level tracking controller and define as *agent model* the closed-loop response of the controlled system. Let $\mathbf{x} \in \mathbb{R}^n$ be the robot state and $\mathbf{u} \in \mathbb{R}^m$ the low-level control input, a generic state model in affine form can be written as

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}, \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^n \times \mathbb{R}^m$ are continuously differentiable functions. The control input in (2) is computed by the robot's low-level controller using the current state and the reference trajectory. Therefore, by employing the parameterization described in Sec. III-B, the controller can be represented as a function $k : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$, i.e.

$$\mathbf{u} = k(t, \mathbf{x}, \mathbf{C}). \tag{3}$$

The agent model is then given by (2) and (3), and the control problem consists of computing the optimized vector of control points $\mathbf{C}$.

D. POIs dynamic model

The vector $\boldsymbol{\xi} = [\xi_1, ..., \xi_M]^{\mathrm{T}} \in \mathbb{R}^M$ contains the heat values associated to the POIs. These values reflect the overall coverage task performance, and robots must minimize its norm. Each element of the vector depends on the distances of the i-th POI from an agent and evolves according to

$$\dot{\xi}_i = h_r(\bar{l}_i) \; + \; \sum_{j=1}^{N} h_d(l_{ij}, \xi_i), \tag{4}$$

where h_r and h_d denote the heat rise and decrement functions, respectively, and the dependence on time is omitted for conciseness. The distance of the i-th point from the nearest robot is given by

$$\bar{l}_i = \min_j l_{ij},$$

where $l_{ij} = \|\boldsymbol{\Pi}\, \mathbf{p}_j - \boldsymbol{\mu}_i\|$ is the Euclidean distance of the i-th POI from the j-th agent's position projected on the ground through the projector $\boldsymbol{\Pi} \in \mathbb{R}^{2\times 3}$.

The rising part is described by

$$h_r(\bar{l}_i) = 2\, K_r^i\, r_i^2\, \bar{l}_i^2\, \operatorname{sech}^2(r_i^2\, \bar{l}_i^2), \tag{5}$$

where $\operatorname{sech}(\cdot)$ is the hyperbolic secant function and K_r^i, $r_i \in \mathbb{R}$ are two constants related to the maximum value of the incremental part. By choosing $r_i = 1/l_i^{max}$, with $l_i^{max} > 0$ being the maximum distance that an agent can attain from the i-th POI, the incremental part in (5) reaches its peak value before the maximum allowable distance from the point is reached, and becomes approximately zero at $l = 2l^{max}$. The values r_i are set by design and can be computed easily given the area size.

The descending part, instead, is maximum when the agent is positioned on the POI and is given by[1]

$$h_d(l_{ij}, \bar{l}_i, \xi_i) = -(K_d^i + \xi_i)\, \frac{1}{2\pi\sigma_i^2}\, \exp\left(-\frac{l_{ij}^2 + \bar{l}_i^2}{2\sigma_i^2}\right), \tag{6}$$

where $\sigma_i(\xi_i) = \sigma_{min} + \tanh^2\left(K_\sigma^i \xi_i\right)(\sigma_{max} - \sigma_{min})$ is the i-th Gaussian distribution variance, $\tanh(\cdot)$ is the hyperbolic tangent function, $K_d^i \in \mathbb{R}$ and $K_\sigma^i \in \mathbb{R}$ are constant gains, and $[\sigma_{min}, \sigma_{max}] \in \mathbb{R}^2$ are minimum and maximum values. Equation (6) defines a decrement term that depends on both the current heat and the agent's position, with its influence decaying exponentially with distance. Moreover, the Gaussian variance—governing the spatial extent over which a robot affects a POI's heat—increases as the heat grows, thus attracting agents. The model in (4) extends the POIs dynamics introduced in [23] to the multi-robot setting by making the heat evolution depend on multiple robots' positions, and reduces to the one in [23] when $N = 1$. In particular, the rise term was designed to depend only on the nearest agent; in this way a single robot can prevent a point from heating up. Conversely, the heat decreases more rapidly when multiple agents converge on the same zone. The use of the minimum distance $\bar{l}_i$ in the decreasing function is not strictly necessary for the coverage task. However, it makes the $i-$th POI heat decrease more sensitive to the $j-$th robot control points variation, if it is the nearest agent. Although this may reduce the efficiency of heat decrease, it promotes better domain separation among agents.

In addition, in [23], the heat was constrained to be strictly positive; here, we allow it to attain negative values by choosing

---

[1] In the case $N = 1$, the term $\bar{l}_i$ is set to zero as it coincides with l_{ij}.

$K_d^i > 0$. As the norm of ξ_i is minimized, agents move away from the point of interest as soon as its value reaches zero. This mechanism encourages agents to move faster to other points of interest or areas on the map; consequently, more erratic motion patterns are generated, as illustrated in Sec. V. For a clearer understanding, the functions' evolutions are depicted in Fig. 3, while further considerations are reported in the Appendix.

Equation (4) depends on all agents' positions and is utilized by the GCS to continuously update the global map using the actual robots' positions. By periodically retrieving updated values, agents maintain coherent POIs' states across the team.

As model (4) describes how the POIs' heats vary on the basis of the current state and the agents' positions, it can be included in the NMPC to predict the heat evolution and optimize motions accordingly. However, each agent optimizes its trajectory in a decentralized way, and it is assumed to be unaware of the remaining robots' motion models. Therefore, single agents can leverage *shared* control points for decentralized optimization. In fact, these points allow each robot to reconstruct other agents' optimized reference trajectories over the horizon and use them instead of the predicted positions. Then (4) can be evaluated by the j-th agent using the distances

$$\begin{aligned} l_{ir}^* &= \|\mathbf{\Pi}\,\mathbf{p}_r^d - \boldsymbol{\mu}_i\|, \quad \forall r \in \{1, \dots, N\} \setminus \{i\} \\ \bar{l}_i &= \min\left\{l_{ij}, \min_r l_{ir}^*\right\}, \end{aligned}$$

where $\mathbf{p}_r^d$ is computed according to (1) using $\mathbf{C}^* = \{\mathbf{c}_{rl},\ l = 0, \dots, 3, \forall r \in \{1, \dots, N\} \setminus \{i\}\}$ that contains the received control points, while $l_{ij} = \|\mathbf{\Pi}\,\mathbf{p}_j - \boldsymbol{\mu}_i\|$ contains the predicted position. In this way, other agents' control points are incorporated within the optimization horizon, enabling the prediction of the POIs' heat evolution due to team-level motions.

E. Optimization problem

This section formulates the NMPC problem solved by the j-th robot, employing the Bézier curve parametrization introduced in Sec. III-B, and the prediction models presented in Sec. III-C and III-D. In the following, the notation $(\cdot)_j$ indicates that the quantity refers to the j-th agent, while the subscript k indicates that it is evaluated at time t_k. Each robot computes the optimized control points by solving the following discrete-time nonlinear problem (NLP)

$$\min_{\mathbf{x}_{j,1},\dots,\mathbf{x}_{j,N_f},\mathbf{C}_j} \quad \sum_{k=1}^{N_f} \|\boldsymbol{\xi}_{j,k}\|_{\mathbf{H}}^2 + J_{j,v} + J_{j,u} + J_{j,r} \tag{7}$$

$$s.t. \quad \text{discretized versions of (2), (3), (4)}$$

$$\mathbf{c}_{j,p} \in A \quad \text{for} \quad p = 0, \dots, 3, \tag{8}$$

$$\mathbf{p}_{j,k} \in A \tag{9}$$

$$\mathbf{x}_{j,k} \in \mathcal{X} \tag{10}$$

$$\mathbf{u}_{j,k} \in \mathcal{U} \tag{11}$$

$$\forall\, k \in \{1, \dots, N_f\},$$

$$\mathbf{p}_{j,0}^d = \mathbf{p}_j(0), \tag{12}$$

$$\mathbf{v}_{j,0}^d = \mathbf{v}_j(0), \tag{13}$$

$$\mathbf{x}_{j,0} = \mathbf{x}_j(0), \quad \boldsymbol{\xi}_{j,0} = \boldsymbol{\xi}(0). \tag{14}$$

where $N_f = T_f/\Delta t$ with Δt being the sampling time, $\mathbf{H} \in \mathbb{R}^{M \times M}$ indicates a positive definite weighting matrix, $\mathcal{X}$ is the state set, $\mathcal{U}$ the input set, and $\mathbf{v}_j$ the robot velocity. The additional terms in the cost function are

$$J_{j,v} = k_v \sum_{k=1}^{N_f} \|\mathbf{v}_{j,k}^d\|^2, \tag{15}$$

$$J_{j,u} = k_u \sum_{k=1}^{N_f-1} \|\mathbf{u}_{j,k}\|_R^2, \tag{16}$$

$$J_{j,r} = k_r \sum_{k=1}^{N_f} \|\mathbf{p}_{j,k}^d - \bar{\mathbf{p}}_{j,k+1}^d\|^2, \tag{17}$$

where $J_{j,v}$ minimizes robot's desired velocities $\mathbf{v}_{j,k}^d$ to avoid bounds' saturation, $J_{j,u}$ is used to minimize the robot's control input, and $J_{j,r}$ is a regularization term, with $\bar{\mathbf{p}}_{j,k+1}^d$ being the previous optimal reference trajectory, which penalizes deviations from the previously computed optimal solution, thereby improving other agents' predictability. Finally, k_v, k_u, k_r are positive gains.

Notice that (12)–(13) are continuity constraints on the reference trajectory and that $\mathbf{p}_{j,0}^d$, $\mathbf{v}_{j,0}^d$ can be expressed in closed form as linear combinations of the control points as

$$\mathbf{p}_{j,0}^d = \mathbf{c}_{j,0}, \tag{18}$$

$$\mathbf{v}_{j,0}^d = \frac{3}{T_f}\left(\mathbf{c}_{j,1} - \mathbf{c}_{j,0}\right), \tag{19}$$

where (18) constraints $\mathbf{c}_{j,0}$ to coincide with the actual robot's position. If smoother accelerations are needed, an additional constraint in the form of

$$\mathbf{a}_{j,0}^d = \bar{\mathbf{a}}_{j,1}^d \tag{20}$$

can be added on initial reference acceleration, where $\bar{\mathbf{a}}_{j,1}^d$ is the previously computed optimal value. Equation (14) imposes initial conditions, using as $\boldsymbol{\xi}_0$ the latest available value received from the central control station. The problem (7)–(14) is solved with period Δt, obtaining the optimized control points; from these, the new position, velocity, and acceleration setpoints can be computed. Notice that the NLP is solved in a receding horizon fashion, sending to the robot's low-level control system only the setpoints evaluated at the next sampling instant. The overall NMPC execution loop is summarized in Algorithm 1.

Algorithm 1 j-th agent NMPC Loop

```
Require: Initial $\mathbf{x}_{j,0}$
Send $\mathbf{c}_{jp} = \mathbf{p}_j\ \forall p = 0, \dots, 3$
Require: first $\mathbf{C}^*$, $\boldsymbol{\xi}_0$,
Initialize state $\mathbf{x}_{j,0} \leftarrow \mathbf{x}_j(0)$,
Initialize heats with values from the GCS $\boldsymbol{\xi}_{j,0} \leftarrow \boldsymbol{\xi}(0)$
while running do
    Update $\boldsymbol{\xi}_{j,0}$ with GCS values, and $\mathbf{x}_{j,0}$,
    if new values then update $\mathbf{C}^*$
    Solve the NMPC optimization problem to obtain $\mathbf{c}_{j,p}^*$, $\forall p = 0, \dots, 3$
    Compute $\mathbf{a}_j^d(\Delta T)$
end while
```

F. Safety

In the NMPC problem (7)–(14), different kinds of constraints can be included to address collision avoidance with other robots. A common choice to avoid inter-robot collisions in coverage tasks is to introduce a buffered Voronoi-based constraint [35]. Taking inspiration from this and leveraging shared control points, half-space constraints can be enforced as

$$\left(\mathbf{p}_i - \frac{\mathbf{p}^d_{r,k} + \mathbf{p}_{i,0}}{2}\right)^T \mathbf{p}_{ri} + r_s\|\mathbf{p}_{ri}\| \leq 0, \, \forall r \neq i, \tag{21}$$

with $\mathbf{p}_{ri} = \|\mathbf{p}^d_{r,k} - \mathbf{p}_{i,0}\|$ and $r_s > 0$ a positive constant. The constraint (21) is imposed along the prediction horizon and is calculated by the $i-$th agent using its measured initial position and the predicted trajectory $\mathbf{p}^d_{r,k}$ of the $r-$th robot, computed from previously shared control points. This choice preserves the constraint linearity in the decision variables and makes the separating plane move along the horizon due to other robots' motions only. Nevertheless, although the task tends to drive agents apart, such constraints can overly restrict robots' motions.

Therefore, we propose a cascaded approach where a safety layer, executed at a higher frequency, allows the system to react to changes in the environment more effectively. This safety layer can be applied either with or without avoidance constraints enforced in the NMPC problem, potentially reducing the computational complexity, at the cost of disregarding them in the planning phase. Specifically, the considered safety layer enforces collision avoidance and boundary constraints by defining appropriate higher-order control barrier functions (HOCBFs) [36] and projecting the reference acceleration onto the safe set through the quadratic problem (QP)

$$\begin{aligned} &\min_{\mathbf{u}} \|\mathbf{u} - \mathbf{a}^d\|^2 && && (22)\\ s.t. \quad &\mathbf{a}_r^{\mathrm{T}}\mathbf{u} \leq b_r \quad \forall r \in \mathcal{N}_r && \textit{(Avoidance)} && (23)\\ &\mathbf{A}_b\mathbf{u} \leq \mathbf{b}_b && \textit{(Boundary)} && (24)\\ &\mathbf{a}_{min} \leq \mathbf{u} \leq \mathbf{a}_{max} && && (25) \end{aligned}$$

where $\mathbf{a}_r$, b_r, $\mathbf{A}_b$ and $\mathbf{b}_b$ can be computed from the CBFs, as in [37], [38], the model, and depend on the measured state, while $\mathbf{a}_{min}$ and $\mathbf{a}_{max}$ are input limits. Finally, the reference acceleration $\mathbf{a}^d$ is obtained by sampling the second derivative of the Bézier curve. The resulting optimization problem is *minimally invasive* [39], convex, and can be solved online at a higher rate than the nonlinear one, possibly employing obstacle measures obtained with onboard sensors, or sampling the reference acceleration at a higher frequency.

In the following simulations and experiments, CBFs have been computed assuming that the other robots' velocity is unknown and thus resorting only to current position values. We highlight that the QP in (22) can also be used to enforce additional safety constraints, such as obstacle/area avoidance, as shown in Sec. V-A, where the avoidance task (23) has been reformulated to operate outside of a no-flight zone.

IV. CASE STUDIES MODELS

To validate the proposed methodology, the nonlinear problem outlined in Sec. III-E is applied to two mobile robot platforms: a flat quadrotor, as uncrewed aerial vehicle (UAV), and a unicycle, as uncrewed ground vehicle (UGV). In both cases, the subscript j, indicating the robot index, is omitted for the ease of notation. These two robots are introduced below and subsequently utilized in various scenarios in Sec. V.

A. Uncrewed Aerial Vehicle

Among UAVs, flat quadrotors utilize reference trajectories to control both position and yaw angle. Since our primary focus is on agents' positions, we neglect the yaw angle: this simplification is justified in cases where the orientation is not critical to the task at hand, such as when a downward-facing camera on a gimbal is mounted.

Introducing the full nonlinear model [40] and the controller expression in (7) would not only complicate the problem, making it challenging to solve online, but also add unnecessary complexity for our purposes; thus, we opt for a simplified approach, assuming that the closed-loop response of the positional part can be described by a double integrator model. Moreover, such a model is suitable for every vehicle not subjected to nonholonomic constraints. Let $\mathbf{x} \in \mathbb{R}^6$ with $\mathbf{x} = \begin{bmatrix}\mathbf{p}^{\mathrm{T}} & \mathbf{v}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$ be the state vector, where $\mathbf{p} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$ are the drone's position and velocity, respectively. The closed-loop model is then given by

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v}, && (26)\\ \dot{\mathbf{v}} &= \mathbf{K}_1\left(\mathbf{p}^{\mathrm{d}} - \mathbf{p}\right) + \mathbf{K}_2\left(\dot{\mathbf{p}}^d - \mathbf{v}\right) + \ddot{\mathbf{p}}^d, && (27) \end{aligned}$$

where $\mathbf{p}^d$, $\dot{\mathbf{p}}^d$ and $\ddot{\mathbf{p}}^d \in \mathbb{R}^3$ are the desired position, velocity, and acceleration vectors, respectively, as computed from (1), while $\mathbf{K}_1 = \mathrm{diag}\left(k_{1,1}, k_{1,2}, k_{1,3}\right)$, $\mathbf{K}_2 = \mathrm{diag}(k_{2,1}, k_{2,2}, k_{2,3})$ are definite positive gain matrices.

In the case of UAVs, if the flight height is not fixed after the optimization, another term in the cost function can account for a nominal flight height z^d

$$J_z = k_z \sum_{k=1}^{N_f}\left(\mathbf{e}_3^{\mathrm{T}}\mathbf{p}^d_{j,k} - z^d\right)^2, \quad \text{with} \quad \mathbf{e}_3 = [0,0,1]^{\mathrm{T}}. \tag{28}$$

B. Uncrewed Ground Vehicle

UGVs are mainly wheeled robots subject to nonholonomic constraints whose kinematic model can be described by

$$\dot{\mathbf{x}} = \left[\dot{x}, \; \dot{y}, \; \dot{\theta}\right]^{\mathrm{T}} = \left[v\cos\theta, \; v\sin\theta, \; \omega\right]^{\mathrm{T}}, \tag{29}$$

where $x, y \in \mathbb{R}$ are the Cartesian coordinates of the contact point of the wheel with the ground, $\theta \in \mathbb{R}$ is the orientation and $v \in \mathbb{R}$ and $\omega \in \mathbb{R}$ are the driving and steering velocities, respectively [41]. Different control laws are available for trajectory tracking [41], both for first and second-order kinematics models. A second-order model (26)–(27) could be used, for example, in the case of a dynamic feedback linearization controller [42]; however, this control approach requires persistent trajectories $(v \neq 0)$ which could not be optimal in the problem here addressed. We therefore assume the presence of a tracking controller based on input/output linearization, obtaining as linear and angular command velocities (26)-(27)

$$v = u_1\cos\theta + u_2\sin\theta, \quad \omega = \frac{u_2\cos\theta - u_1\sin\theta}{b}, \tag{30}$$

with

$$u_1 = \dot{x}_P^d + k_1 \left(x_P^d - x_P \right), \quad u_2 = \dot{y}_P^d + k_2 \left(y_P^d - y_P \right),$$

where $x_P, y_P \in \mathbb{R}$ are the cartesian coordinates of a point P located along the sagittal axis of the unicycle at a distance $|b|$ from the contact point of the wheels on the ground [41], and k_1, k_2 are strictly positive gains. With this choice, (29) corresponds to the robot model (2), equations (30) replace (3), while the term J_u in the cost function is given by

$$J_u = \sum_{k=1}^{N_f} k_u v_k^2 + k_\omega \omega_k^2. \tag{31}$$

Unlike the aerial robot, evaluating the POIs cost function at (x, y), which corresponds to the wheel contact point, may be inefficient for visual monitoring tasks, as UGVs usually mount forward-facing cameras. Thus, a better choice is to evaluate the cost function at P, and plan motions referred to this point.

V. SIMULATIONS

This section presents simulation results in both single-robot and multi-robot scenarios, including cases with homogeneous and heterogeneous teams. The scenarios explored are:

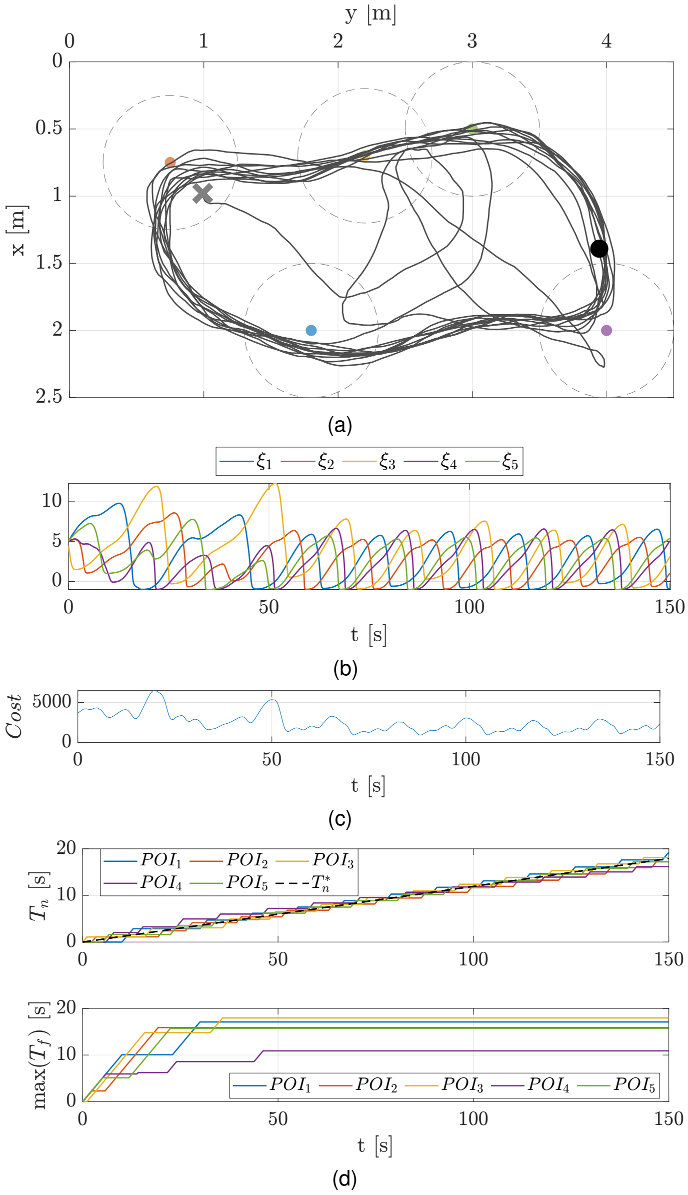
- a single quadrotor,
- a team of a quadrotor and a unicycle,
- a team of three quadrotors.

Additional tests are shown in the accompanying video[2]. Simulations are performed in a ROS2 Humble-Gazebo Garden environment, employing the PX4 firmware software in the loop (SITL) for the quadrotor simulation and the Trutlebot3 model[3]. The nonlinear optimization problems described in Sec. III-E, III-F, IV-A, IV-B are formulated with CasAdi [43] and solved with Ipopt[4] and HSL MA57[5] linear solver. The QP presented in Sec. III-F is instead formulated and solved with the OSQP library [44].

A. Case study: single UAV

The first simulations involve a single drone tasked with covering five points of interest within a map with dimensions $l_x = 2.5$ m $l_y = 4.5$ m, $l_z = 3$ m. The NMPC is solved with $T_f = 1.5$ s and $\Delta t = 0.05$ s for both the control and the prediction steps; the QP is instead solved with sampling time $dt = 0.005$ s. The maximum acceleration is constrained by imposing $|a_{x,y,z}| \leq 2.0$ m/s^2, while the planar velocity $|v_{x,y}| \leq 0.7$ m/s. The gains are $\mathbf{K}_1 = 50\mathbf{I}_3$, $\mathbf{K}_2 = 10\mathbf{I}_3$, $k_z = 1.0$, $k_v = 0.2$, $\mathbf{H} = \mathbf{I}_M$. The regularization term is not employed in the single robot case, while k_u is set to zero. Finally, continuity of acceleration is imposed through (20). The heat dynamics parameters are chosen as $\sigma_{min} = 0.1$, $\sigma_{max} = 0.5$, $\mathbf{K}_r = \mathbf{K}_d = \mathbf{1}_M$, $\mathbf{K}_\sigma = 0.314\mathbf{1}_M$, $\boldsymbol{\xi}_0 = 5\mathbf{1}_M$.

The resulting motion is depicted in Fig. 4a, while Fig. 4b and 4c contain the time evolutions of the heats and of the cost function, respectively. As visible in Fig. 4a, the simulation



(a)

(b)

(c)

(d)

Fig. 4. (a) Path covered by the drone with the initial and final position marked with a cross and a dot, respectively. The POIs are marked with circumferences of radius σ_{max}. (b) Heat values time evolution. (c) NMPC Cost function time evolution. (d) Time evolution of the metric $T_n(i)\ \forall i = 1, \dots, 5$, compared to T_n^* (top); time evolution of the metric $\max(T_f(i))\ \forall i = 1, \dots, 5$ (bottom).

results show the emergence of a period-like motion that covers the five POIs sequentially. This is also evident by evaluating the evolution of $T_n(i)$, $\forall i = 1, \dots, M$, which corresponds to the time spent by the agent near each point (i.e., $\bar{l}_i < \sigma_{max}$). This metric, depicted in Fig. 4d (top), shows a linear increase in time that is comparable with the one given by $T_n^* = t\left(\frac{1}{M} - \frac{T_{min}}{T}\right)$, where $T = 150$ s, $M = 5$, $T_{min} = \frac{L_{min}}{\|\bar{v}_{x,y}\|}$, and indicating with $\|\bar{v}_{x,y}\|$ the norm of the mean planar velocity, and with L_{min} the length of the optimal path connecting all POIs with rectilinear segments. The term $1/M$ corresponds to an equal time division among the POIs, while $\frac{T_{min}}{T}$ accounts for the time needed to travel from one point to another in the optimal visit order—computed by solving a TSP—at the same mean velocity of the simulation. Conversely, the metric $\max(T_f(i))$, defined as the maximum

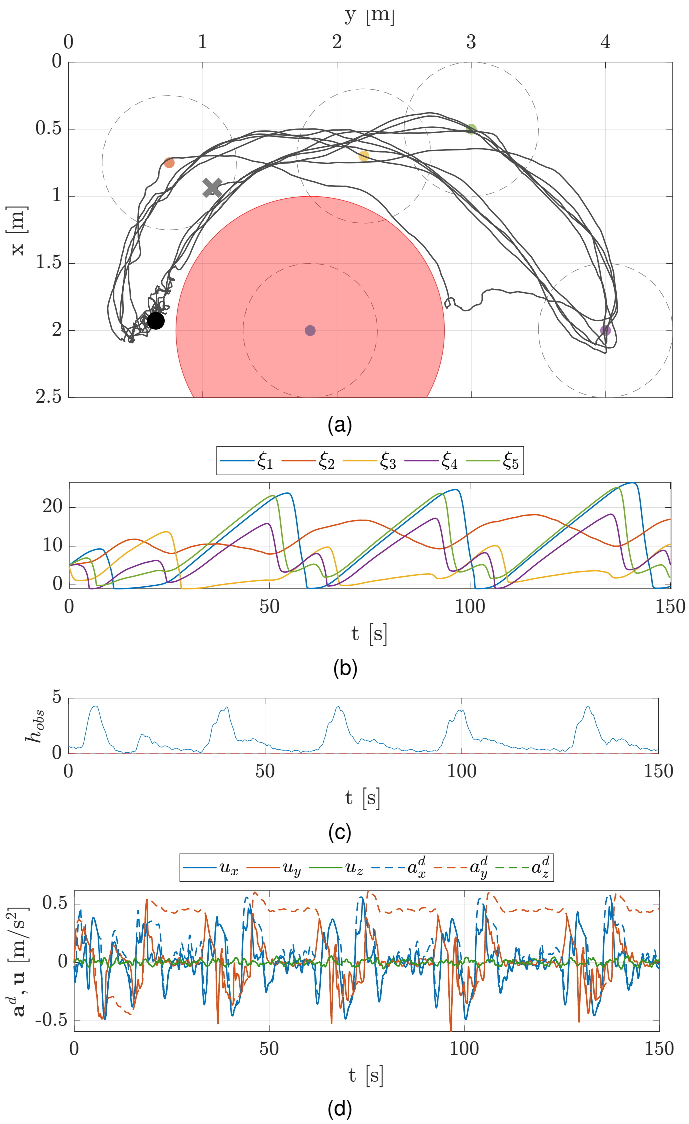
---

[2] https://youtu.be/F9CRAGYzKzo

[3] https://github.com/ROBOTIS-GIT/turtlebot3

[4] https://coin-or.github.io/Ipopt/

[5] https://licences.stfc.ac.uk/product/coin-hsl



(a)

(b)

(c)

(d)

Fig. 5. (a) Path covered by the drone with a no-fly zone indicated by the red circled area. (b) Heat values time evolution. (c) CBF time evolution with $d = 1.0$ m, gains $\gamma_1 = 0.7$, $\gamma_2 = 0.7$. (d) Desired (dashed) and actual (continuous) reference acceleration.

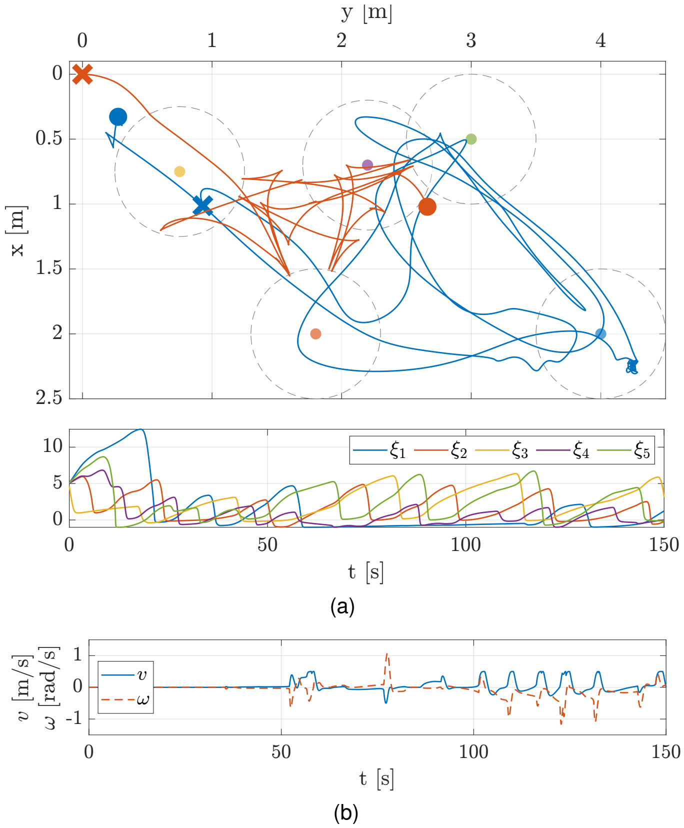
time interval in which the agent is far from the i-th POI ($\bar{l}_i > \sigma_{max}$), converges to a constant value after the transitory, as visible in Fig. 4d (bottom). These results indicate that all points are persistently and effectively monitored over time.

Another simulation is carried out imposing a no-fly zone in the map that occludes a point of interest. This test aims to simulate the impact of an unforeseen operational constraint that interferes with the primary task. The avoidance of this area is obtained only through the safety layer, employing a double integrator model and the CBF

$$h_o(\mathbf{p}) = \|\Pi \mathbf{p} - \mathbf{o}\|^2 - d^2, \tag{32}$$

where $\mathbf{o} \in \mathbb{R}^2$ is the center of the area and d is the minimum distance. Since the relative degree of the system is two, the higher-order CBF can be defined as

$$h_o^1 = \dot{h}_o + \gamma_1 h_o, \tag{33}$$



(a)

(b)

Fig. 6. (a) Path covered by the drone and the rover with the initial and final position marked with a cross and a dot, respectively. The POIs are marked with circumferences of radius σ_{max} (top). Heat values time evolution (bottom). (b) Time evolution of the control inputs v and ω.

and the avoidance task achieved by enforcing the constraint

$$\dot{h}_o^1(\mathbf{p}) \geq -\gamma_2 \left(h^1(\mathbf{p})\right)^3, \tag{34}$$

with γ_1 and γ_2 strictly positive constants. Fig. 5a shows how the no-fly zone modifies the path; notice that the NMPC is unaware of the constraint and that the task performance consequently degrades, as visible in the heats' evolution in Fig. 5c. The CBF value is depicted in Fig. 5b while Fig. 5d shows the desired acceleration references and the safe ones computed from the QP.

B. Case study: UAV and UGV

Simulative results with an unicycle and a drone are presented in the same scenario outlined in the previous section, employing equal coefficients for the POIs' dynamics. The quadrotor NMPC parameters are those employed in the single-robot case, except for $k_r = 0.05$. Unlike the UAV, the unicycle's NMPC is solved with prediction horizon $T_f = 1.5$ s and sampling time $\Delta t = 0.1$ s. It is important to note that the agents are required to share only the prediction horizon value, while they may use different prediction steps due to the parametrization. The parameters used for unicycle's NMPC are $k_v = 40$, $k_\omega = 2.0$ $|v| \leq 0.5$ m/s, $k_r = 0$, $|\omega| \leq 1.5$ rad/s, $k_1 = 2$, $k_2 = 2$, $b = 0.35$ m. Figure 6a shows the path covered by the two robots, without imposing constraints (21), and the time evolution of the POIs values. The rover position

TABLE I
UAV AND UGV

Metric	POI 1	POI 2	POI 3	POI 4	POI 5
T_n/T	0.300	0.310	0.340	0.247	0.204
$\max(T_f)/T$	0.150	0.126	0.140	0.101	0.128

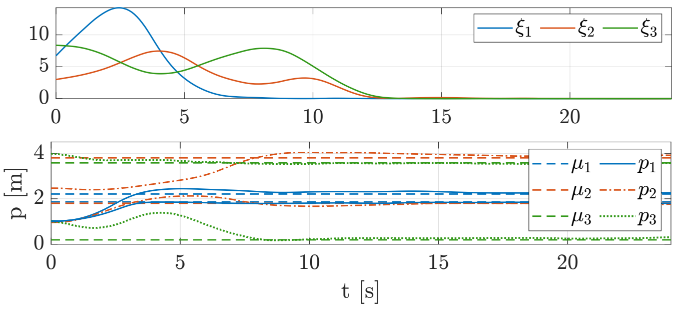


Fig. 7. 3 *UAVs* – 3 *POIs*: ξ values time evolution (top), agents' positions converge to POIs' centers (bottom).

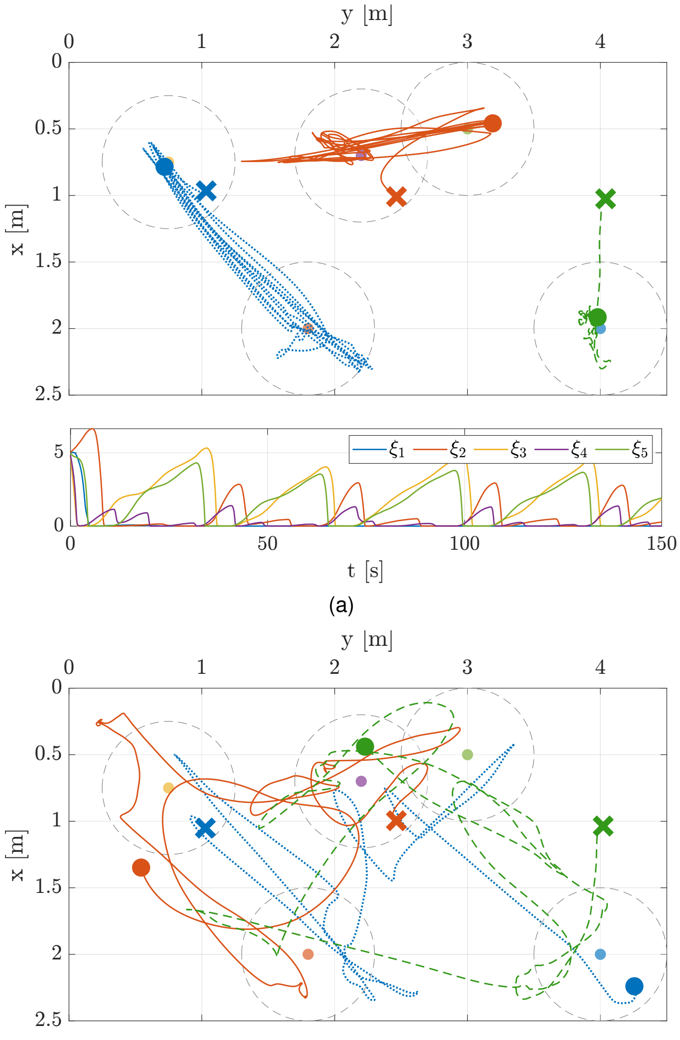
is referred to its center. Figure 6b contains instead the rover control inputs, while the drone's ones are omitted for brevity. Notice that in the reported simulation, the QP layer was not applied to the rover as it was not necessary to address the collision avoidance problem.

Table I reports the final values of the normalized metrics T_n/T and $\max(T_f(i))/T$. Also in this scenario, the agents spend a comparable amount of time near each POI, indicating that all points are covered persistently.

C. Case study: three UAVs

Finally, the results of the three-drones case study are presented. We first show that the proposed algorithm can result in a static deployment in the trivial case of $M = N$, as zero-velocity trajectories can be achieved through coincident control points. Figure 7 illustrates that the heat values are driven to zero and that the agents converge to static deployment positions, in which each agent is exactly over one POI[6]. In this scenario, K_d gains are set to zero as any positive value would prevent the drones from maintaining a static hovering on the POIs' centers. Differently, when the number of agents exceeds the number of points of interest, a subset of robots remains stationary.

Figure 8 depicts instead the drones' motions with $M > N$ and K_d gains set to zero; the figure shows that two agents converge on a periodic motion on a subset of the POIs, while the third remains stationary. The peak values observed in Fig. 8a (bottom) for ξ_3 and ξ_5 are linked to local configurations in which the two agents are equally distant from a POI; since the algorithm runs asynchronously, the agents move either when the heat value exceeds a certain threshold or when another POI's heating breaks the symmetry. However, in none of the simulations performed did the agents get indefinitely blocked in any deadlock configuration.



(a)

(b)

Fig. 8. (a) $\mathbf{K}_d = \mathbf{0}$ – Three UAVs paths with the initial and final position marked with a cross and a dot, respectively (top). Heat values time evolution (bottom). (b) Same scenario with $K_d^i = 1\ \forall i = 1, \ldots, M$.

Differently, Fig. 8b shows the motions obtained with $\mathbf{K}_d = \mathbf{1}_M$. A non-zero value of $\mathbf{K}_d$ acts as a perturbation, as agents are pushed away from the currently covered POI towards nearby ones. This behaviour results in less periodic and more explorative solutions, as visible comparing Fig. 8a and Fig. 8b.

All simulations have been performed without constraint (21) to highlight the motions resulting from the proposed POIs dynamics. In this case study, the inter-robot collision avoidance was obtained only through the safety layer. With the described simulation setup, we obtained a solution time T_s for the NMPC with $\mathrm{mean}(T_s) = 2.2{\cdot}10^{-2}$ s and $\mathrm{std}(T_s) = 7.6{\cdot}10^{-3}$ s, taking the highest value among the three drones.

Finally, Tab. II reports the normalized metrics $T_n(i)$ and $T_f(i)\ \forall i$, and for both values of $\mathbf{K}_d$. From the reported results, we can notice the connection between the considered metrics, and the heat values, as higher peaks of ξ correspond to lower values of T_n and higher T_f. Overall, all points are visited periodically with a coverage homogeneity that may vary depending on the chosen parameters.

---

[6] If there is a subset of narrowly spaced POIs, and their dynamics are slow, one robot could still cover more than one POI.

TABLE II
3 DRONES

	Metric	POI 1	POI 2	POI 3	POI 4	POI 5
$K_d = 0$	T_n/T	0.981	0.592	0.193	0.689	0.272
	$\max(T_f)/T$	0.019	0.064	0.207	0.051	0.173
$K_d = 1$	T_n/T	0.564	0.440	0.231	0.512	0.234
	$\max(T_f)/T$	0.196	0.130	0.206	0.132	0.173

VI. EXPERIMENTS

A. Experimental Setup

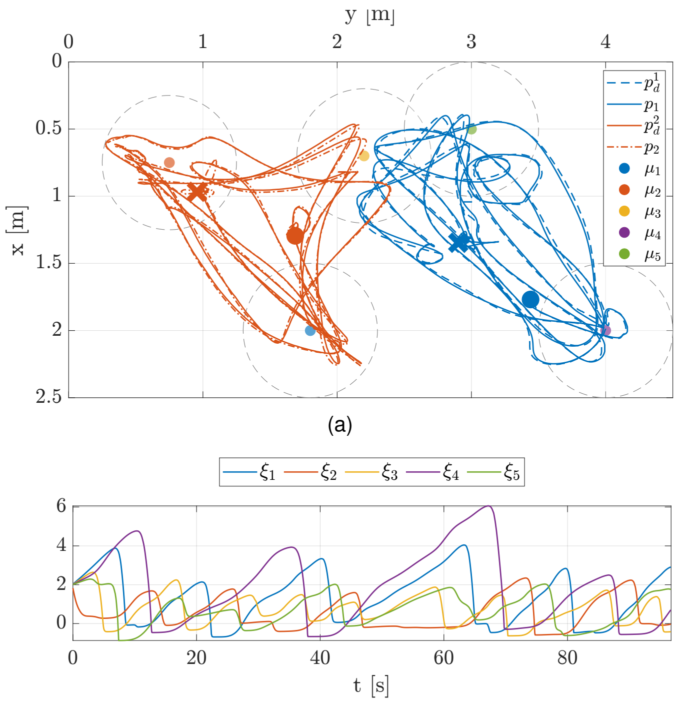
Experiments have been conducted in a flight arena with Optitrack[7] motion capture system employing two quadrotor drones. The platforms utilized Pixhawk 6C and Orange Cube flight controllers with PX4 v1.14 autopilot firmware and a LattePanda 3 Delta[8] as a companion computer. A ROS2 node handled the communication with the PX4 flight controller using the uXRCE-DDS middleware to send trajectory set-points and handle the takeoff and landing phases. All the NMPC communications among agents and with the GCS were performed through ROS2-Humble with a publish-subscribe paradigm, and the optimization problems were solved on the onboard computers. A standard laptop was used instead as central ground station to update the POIs values. Due to the limited arena dimensions, only five POIs were considered.

B. Results

The experimental tests were performed with POIs gains $\mathbf{K}_d = 0.7\mathbf{1}_M$, $\mathbf{K}_r = \mathbf{1}_M$, $\sigma_{min} = 0.1$, $\sigma_{max} = 0.5$, $\mathbf{K}_\sigma = 3.14\mathbf{1}_M$, $\boldsymbol{\xi}_0 = 2\mathbf{1}_M$. The NMPC are solved with $\Delta t = 0.05$ s and $T_f = 1.5$ s, while the QP was solved with $dt = 0.005$ s. The maximum acceleration was constrained by $|a_{x,y,z}| \leq 2.0$ m/s^2, while the velocity is $|v_{x,y}| \leq 0.7$ m/s, $|v_z| \leq 1.0$ m/s. The gains are $\mathbf{K}_1 = 50\mathbf{I}_3$, $\mathbf{K}_2 = 10\mathbf{I}_3$, $k_z = 1.0$, $k_v = 0.2$, $k_r = 0.05$, $\mathbf{H} = \mathbf{I}_M$. The quadratic layer presented in Sec. III-F is used for collision avoidance, while boundary constraints are imposed for additional safety. A planar version of the constraints (21) is imposed.

The resulting paths for both drones are visible in Fig. 9a, where the reference positions are depicted with dashed lines, and the visual odometry ones with continuous lines. Starting locations, indicated with a cross, were chosen naively. The dots correspond to the position at the end of the experiments. Fig. 9a shows that all points are periodically covered by the agents, which approximately equally cover the POIs. Figure 9b depicts the heats' time evolution, showing that the heat values vary periodically over the experiment.

Experimental results demonstrate that the proposed framework performs effectively on standard UAV platforms. Specifically, the optimization problems were solved on CPU-based companion computers, while a commercial autopilot with default parameters was employed as the low-level tracking controller. For additional experiments, refer to the accompanying video.



(a)

(b)

Fig. 9. (a) Desired and actual path with the initial and final position marked with a cross and a dot, respectively. The POIs are marked with circumferences of radius σ_{max}. (b) POIs heats time evolution.

VII. COMPARISON

This section presents a statistical comparison of our approach with the methodology proposed in [16] for a search and rescue application. The latter was selected among the state-of-the-art methods because it implements the most similar dynamic function for the points of interest. More in detail, a potential field is defined in [16] by summing a set of Gaussian radial basis functions for each region of interest. Each function depends on the agent's position and a time-varying reward, which increases the weights of areas that have not been visited recently and decreases those of areas that have. A velocity command is chosen in the direction of steepest descent of this potential field and saturated by a maximum speed. The reader can refer to [16] for a complete overview of the approach. Being based on an artificial potential field, this method will be referred to as PTF from now on.

Statistical analysis is here adopted to discover if significant enhancements in task performance are obtained across control methods in multiple scenarios. Recent papers reported valuable insights thanks to statistical considerations in fields such as robotic manipulation [45] or shared control of robotic systems [46], [47], where variability must be taken into account.

We compared our proposed solution with [16] in the single drone scenario, tuning both functions to obtain similar scaling and dynamics. We consider as factors in our study the control modality, exhibiting two levels (NMPC, PTF), and the number of points of interest M, exhibiting three levels (5, 7, 9). Our aim is to characterize whether a given performance metric has a statistically significant dependence on the considered factors. To this end, we consider the following metrics (T denotes the task duration):

---

[7] https://optitrack.com/

[8] https://www.lattepanda.com/lattepanda-3-delta

- $\mathcal{T}_f = 1/M \sum_{i=1}^{M} \max_i \left(T_f(i)\right)$ is the mean over the POIs of the maximum time intervals $T_f(i)$ in which the $i-$th POI is not covered by the robot (distance $>$ 0.4 m); This metric is representative of how much time the POI can remain uncovered during the task given the control modality and the number of POIs.
- $\mathcal{T}_n = 1/M \sum_{i=1}^{M} T_n(i)/T$, is the mean over the POIs of all time intervals $T_n(i)$ in which the robot is in proximity (distance $\leq$ 0.4 m) of the $i-$th POI; this metric is representative of how well the robotic system is covering the POIs given the control modality.
- $\mathcal{T}_o = \|\mathbf{1}_M/M - T_n/T\|$ is the norm of the difference with respect to an optimal policy in which each POI is covered equally for $1/M$ normalized time. This metric is representative of the covering homogeneity attained by the robotic system under the control modality for the specified number of POIs.
- $\mathcal{V} = 1/T \sum_{t_k=0}^{\mathrm{T}} \|v_{x,y}(t_k)\|$ is the mean in time of the norm of the velocity exhibited by the robot. This metric is representative of the energetic behaviour requested by the robotic system to accomplish the task, given the control modality and the number of POIs.

A balanced two-factor factorial design of the simulations was used to check the significance of factors on the control performance metrics. For each combination of factors, we performed 30 replicates by randomly sampling from a uniform distribution POIs locations over the interval [1,5), and initial $\boldsymbol{\xi}_0$ values over the interval [0, 10).

The results obtained by the series of simulations are reported for visual comparison in terms of box plot (blue), with median values (red line), confidence intervals (black lines), and outliers (red crosses) in Fig. 10. In this figure, each tick of the x-axis represents the given combination of control modality/number of points. To a very close approximation, two estimates being compared are significantly different if their intervals are disjoint, and are not significantly different if their intervals overlap. All the collected data passed the one-sample Kolmogorov-Smirnov test. The analysis of the results is carried out leveraging a two-way ANOVA using a significance level $p = 0.05$. The main effects can be difficult to interpret when the model includes significant interactions between factors, and pairwise comparisons are needed to evaluate the simple main effects of the factors' changes. In our analysis, we have found the statistically significant interactions shown in Fig. 11.

The metric $\mathcal{T}_f$ shows a disordinal interaction ($p < 0.01$, $F = 6.43$) with the trend being inverted when changing the control mode from NMPC to PTF, passing from 5 to 9 POIs. The pairwise comparison revealed a statistically significant change in the metric for the control modality ($p < 0.01$) only when the number of POIs is 5. Instead, for the larger number of POIs the two control modalities perform the same.

The metric $\mathcal{T}_n$ also shows an ordinal interaction ($p = 0.01$, $F = 4.19$) with a greater change in the metric when changing the control mode from NMPC to PTF, passing from 5 to 9 POIs. The pairwise comparison revealed a statistically significant change in the metric for the number of POIs ($p < 0.01$) in both control modalities, showing that the covering time metric $\mathcal{T}_n$ is influenced only by the number of points.

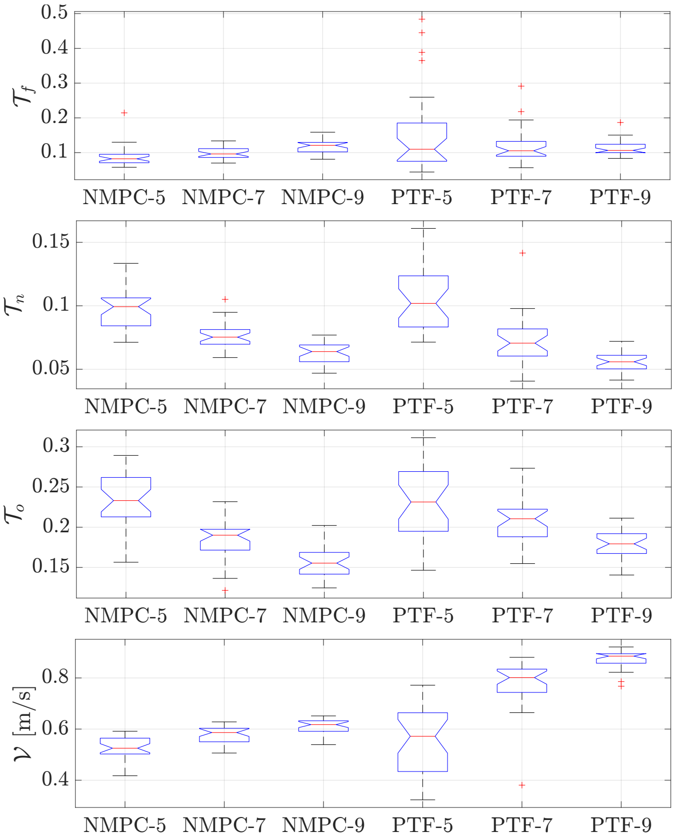


Fig. 10. Visual representation of the statistical evaluation of the results. Left: box plots of the metrics $\mathcal{T}_f$, $\mathcal{T}_n$, $\mathcal{T}_o$, and $\mathcal{V}$ for all the possible combinations of control modality–number of POIs.

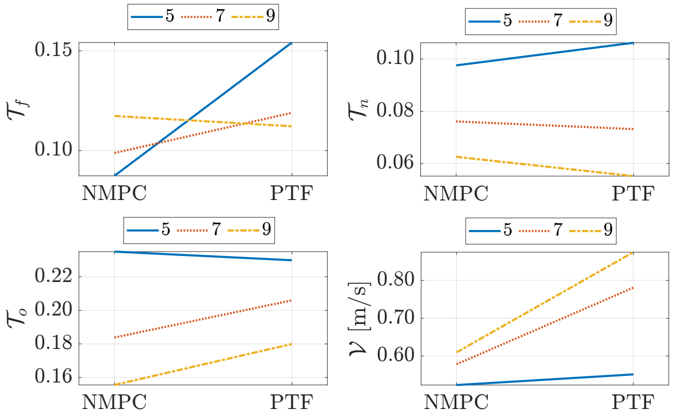


Fig. 11. Interaction plots for the $\mathcal{T}_f$, $\mathcal{T}_n$, $\mathcal{T}_o$, and $\mathcal{V}$ metrics.

The metric $\mathcal{T}_o$ shows an ordinal interaction ($p < 0.01$, $F = 4.61$) with a less marked change in the metric changing the control mode from NMPC to PTF, passing from 5 to 9 POIs. The pairwise comparison revealed a statistically significant change in $\mathcal{T}_o$ for the control modality and the number of POIs except when $M = 5$. This denotes a more homogeneous coverage for NMPC when the number of POIs is large, with equal homogeneity for $M = 5$.

The metric $\mathcal{V}$ shows an ordinal interaction ($p < 0.01$, $F =$

43.03) with a greater difference changing the control mode from NMPC to PTF, passing from 5 to 9 POIs. The pairwise comparison revealed a statistically significant change in $\mathcal{V}$ for the control modality and the number of POIs, except when $M = 5$. This outcome is expected, as the PTF control method's gains were specifically tuned to achieve similar velocities for the case $M = 5$.

The results listed above require special consideration of the existing relationship between the velocity achieved by the robot (expressed by the metric $\mathcal{V}$) and the overall performance (expressed by the remaining metrics). It is worth noting that the notable increase of $\mathcal{V}$ with the number of POIs for the PTF method explains why $\mathcal{T}_f$ does not show a clear increasing trend. Intuitively, by moving faster, the robot will spend less time far from the POIs.

Despite the significant increase in the mean velocity with the number of POIs in the PTF method—always within the imposed limits—the NMPC still performs comparatively or better in the cases highlighted above. It is worth noting that limits imposed on velocity are by design more permissive in the PTF method because we chose to saturate the norm instead of limiting the single components, as done in the NMPC. Thus, we can conclude that adopting the proposed NMPC is advantageous for the considered control objectives, especially with regard to the limited effort required in terms of velocity. Moreover, the NMPC also attains smoother velocities along the task with respect to the PTF approach. These performance improvements arise from integrating both model constraints and POIs' heat evolution in a prediction horizon, a feature that is absent in the gradient-based approach implemented in the benchmark.

VIII. DISCUSSION

The proposed method introduces an innovative approach to persistently monitor a set of POI with a multi-robot system. A key distinction from existing state-of-the-art methodologies lies in the different underlying dynamics, which may render direct comparisons less straightforward as agents move to optimize costs that evolve differently in time. For this reason, task-related metrics were introduced in Sec. VII; however, these metrics were defined by the authors and have not been evaluated in other works.

One limitation of our approach is its focus on local optimality, as global, team-level optimality was not explicitly addressed. However, the use of a decentralized and asynchronous implementation allows greater flexibility. Moreover, currently we cannot exclude the occurrence of deadlocks due to POIs attracting a single robot in opposite directions or multiple robots in the same one. Nevertheless, in the extensive simulation campaign performed in this paper, such conditions were never permanent. Future work could integrate an online deadlock detection mechanism to guarantee escape using, for instance, a prioritization rule.

Additionally, the system architecture is only partially decentralized. While this design choice may limit scalability, the GCS remains essential in most practical scenarios. Moreover, the central unit can retain global information that is otherwise inaccessible from local measurements, as in the application in [13]. However, the proposed architecture has the potential to be fully decentralized, implementing a dynamic consensus algorithm [48] on the heat values, as done in [33].

The NMPC can potentially achieve enhanced task performances by utilizing a higher degree Bézier curve, or multiple, concatenated curves, to allow more complex maneuvers. The impact of this aspect on the computational complexity could be investigated, along with recursive feasibility properties.

Finally, the evolution of the POIs' dynamic depends on the chosen gains; while this aspect allows for significant versatility and adaptability, it requires proper parameter tuning of the heat dynamics if precise task requirements, such as a minimum visit time on the point, must be met. Furthermore, initial heat values can be set according to the monitoring task requirements; for example, points that are more critical in the initial phase can be assigned higher initial heat values.

IX. CONCLUSIONS AND FUTURE WORK

We presented a novel framework for a multi-robot area monitoring problem, considering both homogeneous and heterogeneous teams. Simulations and experimental results demonstrated that the proposed architecture is adaptable to various types of mobile robots and that the optimization problem can be solved online on computationally limited platforms. Moreover, an extensive simulation campaign reported that our NMPC approach outperforms a baseline method in terms of task-related performance and required system velocities in a number of randomly generated scenarios. Future work will focus on removing the central unit by running a dynamic consensus algorithm on the heat values to achieve fully decentralized control. Furthermore, time-varying scenarios, where the number and positions of points of interest are dynamically updated, could be explored. Future work might also investigate the incorporation of energy constraints, such as battery levels, into the CBF layer, as well as explore teams of variable size to enhance energy-aware operations and system scalability.

ACKNOWLEDGEMENTS

The authors wish to thank S. D'Angelo, H. Ullah, and J. Mellet for their help in the experimental phase.

APPENDIX

A. POIs dynamic equation

The heat-like dynamic equation behaviour can be better understood by looking at the function's stationary points in specific cases. If the agent is on the POI's center, fixing $l = 0$, (4) becomes

$$\dot{\xi}_i = f(\xi_i) = -\frac{\left(K_d^i + \xi_i\right)}{2\pi\sigma^2(\xi_i)}, \tag{35}$$

and $\dot{\xi}_i = 0 \iff \xi = -K_d^i$; thus, K_d^i also acts as a lower bound on the values attained by ξ_i. Moreover, the equilibrium is stable if $\sigma(\xi_i)$ is selected properly. The choice of the variance function $\sigma\left(\xi_i\right)$ affects agents' motion and the balance between exploitation and exploration, intended here as the permanence over the current POI or the movement towards a nearby one. Different choices are possible; e.g., by selecting

$$\begin{aligned} \sigma\left(\xi_i\right) &= \sigma_{min}+\tanh^2\left(K_\sigma^i \xi_i\right)\left(\sigma_{max}-\sigma_{min}\right), \\ \sigma\left(\xi_i\right) &= \sigma_{min} \iff \xi_i=0, \end{aligned} \quad (36)$$

the variance also increases as ξ_i becomes negative, pushing an agent away from the POI. Instead, when the agent is at maximum distance $l=l_{max}$ from the POI, (4) becomes

$$\dot{\xi}_i=2\,K_r^i \operatorname{sech}(1)-(K_d^i+\xi_i)\frac{1}{2\pi\sigma_{max}^2}\ \exp\left(-\frac{l_{max}^2}{2\sigma_{max}^2}\right),$$

and the heat upper bound is mainly influenced by the ratio l_{max}/σ_{max}.

References

[1] C. Ercolani *et al.*, "Clustering and Informative Path Planning for 3D Gas Distribution Mapping: Algorithms and Performance Evaluation," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5310–5317, Apr. 2022.

[2] A. Pretto *et al.*, "Building an Aerial–Ground Robotics System for Precision Farming: An Adaptable Solution," *IEEE Robotics & Automation Magazine*, vol. 28, no. 3, pp. 29–49, Sep. 2021.

[3] N. Nigam *et al.*, "Control of Multiple UAVs for Persistent Surveillance: Algorithm and Flight Test Results," *IEEE Trans. Contr. Syst. Technol.*, vol. 20, no. 5, pp. 1236–1251, Sep. 2012.

[4] X. Lin *et al.*, "Robust Planning for Persistent Surveillance With Energy-Constrained UAVs and Mobile Charging Stations," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4157–4164, Apr. 2022.

[5] G. Notomista *et al.*, "A resilient and energy-aware task allocation framework for heterogeneous multirobot systems," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 159–179, 2022.

[6] M. Coffey *et al.*, "Covering Dynamic Demand with Multi-Resource Heterogeneous Teams," in *2023 IEEE/RSJ Int. Conf. Intell. Rob. Syst. (IROS)*, Oct. 2023, pp. 11 127–11 134.

[7] M. Santos *et al.*, "Coverage Control for Multi-Robot Teams with Heterogeneous Sensing Capabilities Using Limited Communications," in *IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, Oct. 2018, pp. 5313–5319.

[8] S. Gao *et al.*, "Effective Dynamic Coverage Control for Heterogeneous Driftless Control Affine Systems," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2018–2023, Dec. 2021.

[9] A. Mavrommati *et al.*, "Real-Time Area Coverage and Target Localization Using Receding-Horizon Ergodic Exploration," *IEEE Trans. Robot.*, vol. 34, no. 1, pp. 62–80, Feb. 2018.

[10] Y. Liu *et al.*, "Robotic urban search and rescue: A survey from the control perspective," *J. Intell. Robot. Syst.*, vol. 72, 11 2013.

[11] A. Ghotavadekar *et al.*, "Variable time-step mpc for agile multi-rotor uav interception of dynamic targets," *IEEE Robot. Autom. Lett.*, vol. 10, no. 2, pp. 1249–1256, 2025.

[12] O. Archila *et al.*, "Optimal distribution of uavs in crop spraying considering energy consumption," in *2024 Am. Control Conf.*, 2024, pp. 2023–2028.

[13] R. Caccavale *et al.*, "A multi-robot deep q-learning framework for priority-based sanitization of railway stations," *Applied Intelligence*, vol. 53, pp. 1–19, 04 2023.

[14] F. Nekovář *et al.*, "Multi-vehicle dynamic water surface monitoring," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6323–6330, 2023.

[15] K. Jakkala *et al.*, "Probabilistic Gas Leak Rate Estimation Using Submodular Function Maximization With Routing Constraints," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5230–5237, Apr. 2022.

[16] J. R. Cooper, "Optimal Multi-Agent Search and Rescue Using Potential Field Theory," in *AIAA Scitech 2020 Forum*. Orlando, FL: American Institute of Aeronautics and Astronautics, Jan. 2020.

[17] S. K. K. Hari *et al.*, "Optimal uav route planning for persistent monitoring missions," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 550–566, 2021.

[18] M. Ostertag *et al.*, "Trajectory planning and optimization for minimizing uncertainty in persistent monitoring applications," *J. Intell. Robot. Syst.*, vol. 106, 08 2022.

[19] C. Song *et al.*, "Optimal control for multi-agent persistent monitoring," *Automatica*, vol. 50, no. 6, pp. 1663–1668, 2014.

[20] M. Saska *et al.*, "Autonomous deployment of swarms of micro-aerial vehicles in cooperative surveillance," in *2014 Int. Conf. on Unmanned Aircraft Systems*, 2014, pp. 584–595.

[21] N. Nigam *et al.*, "Control of multiple uavs for persistent surveillance: Algorithm and flight test results," *IEEE Trans. Contr. Syst. Technol.*, vol. 20, no. 5, pp. 1236–1251, 2012.

[22] T. Chung *et al.*, "Search and pursuit-evasion in mobile robotics," *Auton. Robots*, vol. 31, 11 2011.

[23] S. Marcellini *et al.*, "Nonlinear model predictive control for repetitive area reconnaissance with a multirotor drone," in *2023 Int. Conf. on Unmanned Aircraft Systems*, 2023, pp. 515–522.

[24] J. Cortes *et al.*, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[25] Y. Bai *et al.*, "Safe adaptive multi-agent coverage control," *IEEE Control Systems Letters*, vol. 7, pp. 3217–3222, 2023.

[26] R. Funada *et al.*, "Visual coverage control for teams of quadcopters via control barrier functions," in *2019 Int. Conf. Robot. Autom.*, 2019, pp. 3010–3016.

[27] M. Santos *et al.*, "Decentralized minimum-energy coverage control for time-varying density functions," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems*, 2019, pp. 155–161.

[28] A. Carron *et al.*, "Model predictive coverage control," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6107–6112, 2020.

[29] Y. Wang *et al.*, "Awareness coverage control over large-scale domains with intermittent communications," *IEEE Transactions on Automatic Control*, vol. 55, no. 8, pp. 1850–1859, 2010.

[30] C. Song *et al.*, "Decentralized adaptive awareness coverage control for multi-agent networks," *Automatica*, vol. 47, no. 12, pp. 2749–2756, 2011.

[31] ——, "Persistent awareness coverage control for mobile sensor networks," *Automatica*, vol. 49, no. 6, pp. 1867–1873, 2013.

[32] E. J. Rodríguez-Seda *et al.*, "Decentralized persistent area coverage control with loss of awareness," in *2020 IEEE Conference on Control Technology and Applications (CCTA)*, 2020, pp. 528–535.

[33] X. Xu *et al.*, "Persistent awareness-based multi-robot coverage control," in *2020 59th IEEE Conf. Decis. Control.*, 2020, pp. 5315–5320.

[34] T.-H. Cheng *et al.*, "Awareness coverage control with uncertain loss of awareness," in *2018 Annual Am. Control Conf.*, 2018, pp. 4912–4917.

[35] D. Zhou *et al.*, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 1047–1054, 2017.

[36] W. Xiao *et al.*, "Control barrier functions for systems with high relative degree," in *IEEE 58th Conf. Decis. Control.*, 2019, pp. 474–479.

[37] L. Wang *et al.*, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 661–674, 2017.

[38] S. Wilson *et al.*, "The robotarium: A remotely-accessible, multi-robot testbed for control research and education," *IEEE Open Journal of Control Systems*, vol. PP, pp. 1–13, 01 2022.

[39] A. D. Ames *et al.*, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference*, 2019, pp. 3420–3431.

[40] D. Mellinger *et al.*, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE Int. Conf. Robot. Autom.*, 2011, pp. 2520–2525.

[41] B. Siciliano *et al.*, *Robotics: Modelling, Planning and Control*, ser. Advanced Textbooks in Control and Signal Processing. Springer London, 2010.

[42] A. De Luca *et al.*, "Stabilization of the unicycle via dynamic feedback linearization," *IFAC Proceedings Volumes*, vol. 33, no. 27, pp. 687–692, 2000.

[43] J. A. E. Andersson *et al.*, "CasADi – A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.

[44] B. Stellato *et al.*, "OSQP: an operator splitting solver for quadratic programs," *Math. Program. Comput.*, vol. 12, no. 4, pp. 637–672, 2020.

[45] V. Morlando *et al.*, "Nonprehensile object transportation with a legged manipulator," in *2022 Int. Conf. Robot. Autom.*, 2022, pp. 6628–6634.

[46] M. Selvaggio *et al.*, "Shared-control teleoperation methods for a cable-suspended dual-arm unmanned aerial manipulator," in *2024 Int. Conf. on Unmanned Aircraft Systems*, 2024, pp. 1132–1139.

[47] ——, "A shared-control teleoperation architecture for nonprehensile object transportation," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 569–583, 2022.

[48] S. S. Kia *et al.*, "Tutorial on dynamic average consensus: The problem, its applications, and the algorithms," *IEEE Control Systems Magazine*, vol. 39, no. 3, pp. 40–72, 2019.