Route Optimization in Precision Agriculture Settings: A Multi-Steiner TSP Formulation

Antonio Furchì, Martina Lippi, *Associate Member, IEEE*, Renzo Fabrizio Carpio, and Andrea Gasparri, *Senior Member, IEEE*

***Abstract*— In this work, we propose a route planning strategy for heterogeneous mobile robots in Precision Agriculture (PA) settings. Given a set of agricultural tasks to be performed at specific locations, we formulate a multi-Steiner Traveling Salesman Problem (TSP) to define the optimal assignment of these tasks to the robots as well as the respective optimal paths to be followed. The optimality criterion aims to minimize the total time required to execute all the tasks, as well as the cumulative execution times of the robots. Costs for travelling from one location to another, for maneuvering and for executing the task as well as limited energy capacity of the robots are considered. In addition, we propose a sub-optimal formulation to mitigate the computational complexity by leveraging the fact that generally in PA settings only a few locations require agricultural tasks in a certain period of interest compared to all possible locations in the field. A formal analysis of the optimality gap between the optimal and the sub-optimal formulations is provided. The effectiveness of the approach is validated in a simulated orchard where three heterogeneous aerial vehicles perform inspection tasks.**

***Note to Practitioners*—This paper aims at providing an efficient solution to PA needs by deploying a team of robots able to perform agricultural tasks at given locations in large-scale orchards. In particular, a novel general optimization problem is proposed that, given a set of mobile and possibly heterogeneous robots and a set of agricultural tasks to carry out, defines the assignment of these tasks to the robots as well as the routes to follow, while minimizing the total and the cumulative execution times of the robots. Existing approaches for route optimization in PA generally involves complete coverage of the field by one or multiple robots and do not account for maneuvering costs with general layouts of the field. We consider costs for travelling from one location to another, for executing the task and for maneuvering without any restriction on the layout of the plants as well as we take into account the limited energy capacity of the robots. We also provide a sub-optimal formulation which reduces the computational burden by relaxing the optimization of the maneuvering costs at the locations where agricultural tasks are carried out and formally derive the optimality gap. The proposed approach is flexible and can be easily adapted to any PA setting involving multiple mobile robots that are required to accomplish given tasks in an area of interest. We validate its effectiveness in a realistic simulated setup composed of three heterogeneous aerial vehicles performing inspection tasks. In future research, we aim to design algorithms to solve the proposed optimization problems in an efficient manner as well as to validate the formulations on real-world robotic platforms.**

***Index Terms*— Multi-robot systems, route optimization, precision agriculture.**

I. Introduction

CONTINUOUS plant-by-plant monitoring and targeted interventions are key features of the Precision Agriculture (PA) paradigm, that potentially enable increased crop productivity while reducing waste. Although monitoring activities can generally be carried out through remote sensing [1], close-up operations, which we will refer to as *agricultural tasks*, are typically required for intervention activities such as pesticide [2], herbicide [3] and fertilizer [4] release, inspection [5], weed detection [6], pruning [7], or harvesting [8]. Deploying multiple, and possibly heterogeneous, mobile robots in the field, which autonomously navigate among plants [9] and carry out agricultural tasks, provides an effective solution to this proximity need [10]. In this way, robots can reach out to the different plants requiring intervention and perform the latter in a parallel fashion. However, realistically, in large-scale PA settings, agricultural tasks may be required in locations that are sparse with respect to the size of the field and low in number compared to the total number of plants. This is motivated by the fact that the needs of the plants in the field can be highly variable due to the varietal assortment, e.g., [11], and/or different soil or climate conditions, e.g., [12]. Based on the above, a fundamental question to be addressed is to define which robot should treat which plants by wisely choosing the respective path to be followed to reach them.

For this purpose, appropriate optimization metrics must be defined which are capable of taking into account for the overall execution times by the robots.

Moreover, it is also generally desirable to mitigate as much as possible the occurrence of maneuvering or turning operations in the planned paths since these are typically costly in terms of both time and energy, requiring, for example, high torques to the motors to be executed. Finally, the limited energy capacity of the robots should also be taken into account

Manuscript received 17 April 2022; revised 3 July 2022; accepted 11 July 2022. This article was recommended for publication by Associate Editor M. Franceschelli and Editor B. Vogel-Heuser upon evaluation of the reviewers' comments. This work was supported in part by the European Commission, Project PANTHEON, under Agreement 774571; in part by the Regione Lazio within the Programs POR FESR Lazio 2014-2020, Avviso Pubblico "Progetti di Gruppi di Ricerca 2020," AGR-o-RAMA, under Agreement B85F21001360006; and in part by POR FESR Lazio 2014-2020, Avviso Pubblico "Progetti Strategici," Project Paradise, under Agreement F84D20000030003. *(Corresponding author: Martina Lippi.)*

The authors are with the Dipartimento di Ingegneria, Università degli Studi Roma Tre, 00146 Rome, Italy (e-mail: martina.lippi@uniroma3.it).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TASE.2022.3204584.

Digital Object Identifier 10.1109/TASE.2022.3204584

since charging stations may be few and not always easily accessible in agricultural environments.

Motivated by the reasons mentioned above and driven by the needs of some research projects on the use of robotics in PA settings, we devise a flexible planning strategy which, given a set of heterogeneous mobile robots and a set of agricultural tasks to be performed, defines the tasks that are assigned to each robot and the respective paths to carry out the assigned agronomic interventions. As envisioned in Supervisory Control and Data Acquisition (SCADA) architectures, which can be applied in PA settings as in the the H2020 PANTHEON project, we consider that a central unit gathers data from the field and is in charge of computing optimal routes for the robots. In this work, firstly, we define a field graph model, denoted as *orientation* graph, that accounts for turning costs with any planting pattern, i.e., any layout of plants in the field. Secondly, we formalize a multi-Steiner Traveling Salesman Problem (TSP), which extends the Steiner TSP to the multi-robot case. Turning costs as well as energy capacity constraints are taken into consideration in the proposed formulation. Thirdly, we provide an additional sub-optimal formulation which is based on the observation that, in a large-scale field, only few locations compared to the large number of possible ones typically require agronomic interventions, as discussed above due to the varietal assortment, e.g., [11], and/or different soil or climate conditions, e.g., [12]. The optimization of the turning costs is thus relaxed at these locations to reduce the computational burden at the price of an optimality gap. This gap has been theoretically investigated and a bound has been formally derived. Finally, simulation results on a realistic PA setting composed of three aerial robots show the effectiveness of the proposed two formulations and the related behaviors. An illustrative example of the envisioned PA setting is shown in Figure 1 in which three aerial robots have to perform inspection tasks on specific plants in a hazelnut orchard. Note that the proposed formulation is not limited to inspection tasks in PA settings, but could be applied to any setting where optimal routes for multiple robots to perform tasks in assigned locations of the environment must be computed. This includes, for example, logistics applications. In addition, in PA-specific settings, it can be adapted to carry out any agricultural task in permanent crops, e.g., release of herbicide or pesticide, pruning or harvesting.

This paper builds on [13] with respect to which the following contributions are introduced:

- A novel field modeling is defined which allows to take into account maneuvering costs with general field topology. Moreover, three-dimensional orchards are modeled in which also the height elevation is considered making the formulation suitable for both aerial and ground robots.
- An extended optimization problem is formulated where the maximum execution time and the cumulative execution times by the robots are minimized and possible limited operational capabilities of the robots over time, due to the limited energy capacity, are taken into consideration. Moreover, teams composed by heterogeneous robots, i.e., robots with different energy capacities and/or temporal costs, are enabled.

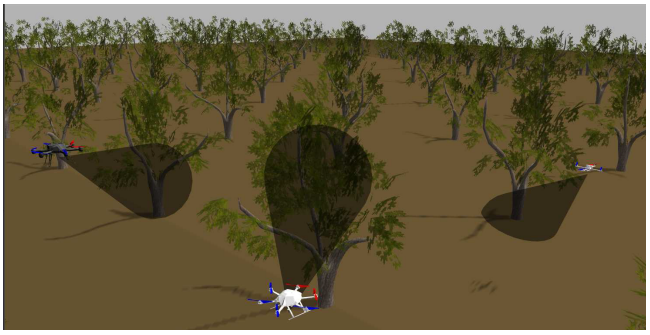


Fig. 1. Example of operating scenario in which three heterogeneous aerial robots are deployed in a hazelnut orchard for targeted inspection tasks.

- A sub-optimal formulation is proposed boosting the solution computation and a formal analysis of the optimality gap is provided.
- A greedy solution is designed and implemented to compare the performance of the proposed formulations.

The remainder of the paper is organized as follows. First, relevant works at the state of the art are discussed and compared to the proposed approach in Section II. Then, the multi-Steiner TSP for PA settings is formalized in Section IV and a sub-optimal formulation to reduce the computational load is proposed in Section V. Finally, simulation results with three aerial vehicles are presented in Section VI.

II. Related Work

A typical problem in PA settings is how to optimally perform complete *coverage* of the field by one or multiple mobile robots, i.e., how to find the routes associated to each robot in order to visit *all* the locations of the field while optimizing a certain objective function. The field is generally modeled as a set of parallel tracks, e.g., [33], [34], [35], [36]. In this work, we are not interested in solving a coverage problem but rather we aim to define optimal routes for multiple robots, with limited energy capacity, to visit only a *subset* of all possible locations, as pointed out in the Introduction. Moreover, we aim to consider a general field topology, meaning that it can accommodate any planting pattern, while taking into account maneuvering costs and limited energy capacity.

In the context of route optimization, the Traveling Salesman Problem [47] lays the foundation for performing coverage when a single mobile agent is involved: given a set of target locations, the TSP aims to find the shortest path that crosses all of them, returning to the origin location. Note that the TSP formulation and its variants presented in the following do not fulfill the requirements of our work, since they focus on a single robot scenario covering the entire field, instead of target locations. As instance for TSP in PA contexts, a relevant work can be found in [14] which considers a spraying application by an Unmanned Aerial Vehicle (UAV). Stressed areas in the field are first identified and, then, the shortest path for traversing them is defined using a TSP-based solution. Similarly, the identification of stressed areas is considered in [15] where a TSP routing algorithm is combined with human inputs to build a collaborative and adaptive framework. A TSP-based

TABLE I

OVERVIEW OF PROBLEMS CONSIDERED BY STATE OF THE ART PAPERS: TSP, STEINER TSP, MULTI-TSP, VRP, OP, GMDTSP AND MULTI-STEINER TSP. THE WORKS IN THE TABLE ARE DESCRIBED IN DETAIL IN SECTION II AND, FOR THE SAKE OF READABILITY, THEY ARE INDEXED BY THEIR REFERENCE NUMBER

Related work	TSP	STSP	mTSP	VRP	OP	GMD TSP	mSTSP
[14][15] [16][17] [18][19] [20][21] [22][23] [24][25] [26]	✓						
[27]		✓					
[28][29] [30][31] [32]			✓				
[33][34] [35][36] [37][38] [39][40] [41][42]				✓			
[43][44]					✓		
[45]						✓	
[13][46] Ours							✓

approach is also devised in [16] for releasing pesticides in pests-ridden areas. Possible obstacles between the field areas are taken into account. Pesticide application is additionally addressed in [17] where a TSP-based formulation is designed relying on a Model Predictive Control-based demand management to define the optimal amount of pesticide to release in each location. Furthermore, the work in [18] combines the TSP with the coverage path planning to survey several regions that are spatially distributed by a UAV. Finally, path planning for a UAV in PA wireless sensor networks is also addressed in [19] where a heuristic model for the TSP is exploited.

Variants of the TSP have been studied to address monitoring problems for a UAV in contexts beyond PA [20], [21], [22], [23], [24], [25], [26]. In these problems, particular attention is generally paid to the UAV's energy supply, which is necessary for the success of the mission. More in detail, in [20] the UAV can recharge in dedicated depots and the route minimizing the fuel consumption, while visiting all the locations and complying with energy constraints, is computed. The possibility of recharging on Unmanned Ground Vehicles (UGVs), moving with same or lower velocity than the UAV, is introduced in [21] and efficient solving algorithms are provided in [22]. A similar scenario is also considered in [23], tackling different velocities for the aerial and ground robots. A particular model of the field using boustrophedon cells is used in [24], where a coverage problem using a UAV, which can land on a UGV to refuel, is addressed. In the context of PA, a boustrophedon cell can represent a row of a crop. Differently from the above methods, the work in [25] considers a coverage problem with a UAV-UGV team acting as a unique entity: the UAV travels on the UGV and leaves only to reach locations that are inaccessible to the UGV. Persistent monitoring is studied instead in [26] which defines a min–max weighted latency walk problem, where the robot is required to repeatedly carry out a closed walk in a graph with weighted vertices.

An extension of the classical TSP is the *Steiner* TSP in which only a subset of all the possible locations is required to be visited. However, as per the TSP, also this formulation only considers a single robot and thus it is not appropriate for our setting. As instance of Steiner TSP, the study in [27] generalizes the Steiner TSP to visit each target location twice and requires that a human operator provides instructions about the agricultural tasks before their execution. Possible communication limitations are tackled.

A further extension of the TSP enables the inclusion of multiple robotic platforms. In this case, a multi-TSP [48] is formulated, which however does not foresee to only visit target locations in the field, as we require, and does not take into account limited energy capacity of the robots. An application of multi-TSP to harvesting in PA settings can be found in [28] in which travel distance and workload balancing among the vehicles are optimized. A monitoring problem using UAVs or UGVs is addressed in [29] where two path planning problems to visit all the possible locations are defined for UAVs and UGVs, respectively. In both problems, the robots in the team are heterogeneous and each robot can visit different subsets of nodes. Persistency in coverage problems with multiple robots is addressed in [30], [31], [32]. In particular, in [30], multiple UAVs are in charge of monitoring the environment while multiple UGVs are in charge of providing recharge to the UAVs. The authors assume that the paths of the UAVs are assigned and the objective is to plan the routes for the UGVs accordingly. An extension with efficient algorithms is provided in [31]. A team composed of homogeneous UAVs only is considered instead in [32] to realize persistent coverage of target nodes. The goal of this work is to determine cycles for UAVs such that the data of all target nodes is collected and sent to a control station within a desired frequency and with a maximum delivery time.

More commonly, when multiple mobile robots are involved in the system, a Vehicle Routing Problem (VRP) [49] is set-up in which also limited capacities of the robots are taken into account compared to the multi-TSP. However, as for the multi-TSP, all the locations of the field are visited in a VRP which is not appropriate to our setup. Works in [33] and [34] formulate a VRP in PA settings in which the field is modeled through parallel tracks and each track is associated with a node in a graph. All the nodes are required to be crossed by exactly one vehicle, while not exceeding the vehicle capacity. Routes are obtained minimizing the headland turning costs from the end of one track to the start of the following one. Similar formulations can be found in [37], where intra- and inter-row orchard operations are possible, in [38], where the presence of possible obstacles in the tracks is introduced, in [36], where two kinds of turns are characterized, in [39], where VRP-based solutions are compared to traditional routes performed by expert human drivers, in [40], where an Evolutionary Hybrid Neighbourhood Search algorithm is proposed to solve the VRP, and in [35], where no turning costs are taken into account but an adaptive large neighborhood search-based solution is provided to find the robots' routes in an efficient manner.

A more general topology of the field is then considered in [41] where the latter is modeled as a square grid graph. A VRP with turning penalties is formulated and a minimum cost network flow problem is defined to improve the computational times. Persistency in coverage problems for heterogeneous UAVs, having limited energy capacity, and UGVs with restricted visibility is then studied in [42].

An Orienteering Problem (OP), that is a variant of the TSP, is then formulated in [43] where a UAV is exploited to retrieve aerial measurements near points that are potentially mislabeled. The problem of maximizing the number of points visited by the UAV while complying with the limited capacity is addressed. A *Team* OP, that is a variant of the VRP, is considered in [44] where an irrigation application in vineyards is considered. Multiple ground robots need to traverse a planar graph where nodes, associated with water emitters, have rewards, and edges have costs and are associated with possible paths between emitters. The paths maximizing the sum of rewards for visited nodes while complying with limited capacities of the robots are found and heuristics are defined.

Another extension of the TSP is the Generalized multi-depot TSP (GMDTSP) [45], in which the target locations are partitioned into clusters and the overall shortest paths for the salesmen to visit at least one target location for each cluster, starting from distinct depots, are found. However, the GMDTSP does not allow to visit only specific locations as we require.

Differently from the papers described above, we propose a *multi-Steiner TSP* extending the Steiner TSP to *i) multiple* heterogeneous mobile robots and *ii)* allowing to visit only a subset of all possible locations within the field, as typical for PA settings. To summarize, as reported above, existing formulations do not provide a viable solution accounting for both points *i)* and *ii)*, i.e., the simple TSP requires a *single* robot to visit *all* the possible locations, the Steiner TSP requires a *single* robot to visit a subset of locations, the multi TSP requires multiple robots to visit *all* possible locations, the vehicle routing requires multiple robots with limited capacity to visit *all* the locations, while the GMDTSP requires multiple robots to visit all *clusters*, so that at least one node for each cluster is visited by at least one robot. It is worth mentioning that a multi-Steiner TSP is also addressed in [46]. However, a fundamental difference compared to our contribution is that the work in [46] assumes that the set of nodes to serve is *pre-assigned* to each robot. In contrast, we here aim to determine the *optimal* set of nodes to assign to each robot as an *output* of the optimization problem.

Table I summarizes the above mentioned works at the state of the art, categorizing them according to the addressed problems, i.e., TSP, Steiner TSP, multi-TSP, VRP, OP, GMDTSP and multi-Steiner TSP. This paper advances the state of the art on the following aspects:

i) A comprehensive multi-Steiner TSP formulation is proposed;
ii) A novel field model is devised which allows to tackle turning costs with any field topology, i.e., orchards are not restricted to exhibit only parallel tracks or regular planting pattern;
iii) A sub-optimal formulation is proposed and formally analyzed to reduce the computational burden.

TABLE II
MAIN NOTATIONS INTRODUCED IN THE PAPER

Variable	Meaning
$\mathcal{B}$	Binary set $\{0,1\}$
$\mathcal{G}=(\mathcal{V},\mathcal{E})$	Field graph with vertices $\mathcal{V}$ and edges $\mathcal{E}$
$\mathcal{R}$	Set of robots
$\mathcal{V}_s$	Set of vertices to serve
δ_i^+ (δ_i^-)	Set of edges having vertex v_i as head (tail)
$\mathcal{E}_c$	Set of consecutive edges
p_i	3-dimensional location of the i th vertex
$\varphi_{p,q}$	Angle between consecutive edges e_p and e_q
l_k	Length of the edge e_k
$\bar{\mathcal{G}}=(\bar{\mathcal{V}},\bar{\mathcal{E}})$	Orientation graph with vertices $\bar{\mathcal{V}}$ and edges $\bar{\mathcal{E}}$
$\bar{\xi}_i^+$ ($\bar{\xi}_i^+$)	Set of edges in $\bar{\mathcal{E}}$ which flow into (originate from) $v_i \in \mathcal{V}$ of the field graph
$x_{k,j}$	Decision variable which is 1 if robot r_j passes through edge $\bar{e}_k$, 0 otherwise
$s_{i,j}$	Decision variable which is 1 if robot r_j serves node $v_i \in \mathcal{V}_s$, 0 otherwise
K_j	Energy capacity of robot r_j
$c_j^e(c_j^t)$	Unit temporal cost for traversing an edge (turning) by robot r_j
$c_{i,j}^s$	Temporal cost for serving node v_i by robot r_j
$\varepsilon_j^e(\varepsilon_j^t,\varepsilon_j^s)$	Unit energy cost of robot r_j when traversing an edge (turning, serving a node)
$f_{k,j}$	Commodity variable for robot r_j through edge e_k
$\hat{\mathcal{G}}=(\hat{\mathcal{V}},\hat{\mathcal{E}})$	Service graph with vertices $\hat{\mathcal{V}}$ and edges $\hat{\mathcal{E}}$
$\hat{(\cdot)}$	Variables referred to the service graph $\hat{\mathcal{G}}$
$\hat{L}_{h,j}=(\hat{L}_{h,j}^e,\hat{L}_{h,j}^t)$	Tuple associated with edge $\hat{e}_h=(v_s,v_t)\in\hat{\mathcal{E}}$ and robot r_j comprising the overall temporal costs to traverse the edges from v_s to v_t ($\hat{L}_{h,j}^e$) and for turning operations ($\hat{L}_{h,j}^t$)
$a_{k,h,j}$	Decision variable for the local problem which is 1 if edge $\bar{e}_k\in\bar{\mathcal{E}}$ belongs to the shortest path from v_s to v_t for robot r_j, 0 otherwise
$\mathcal{P}_{h,j}$	Path associated with edge $\hat{e}_h=(v_s,v_t)\in\hat{\mathcal{E}}$ and robot r_j minimizing the temporal costs from v_s to v_t
$\hat{x}_{k,j}$	Decision variable for the sub-optimal problem which is 1 when robot r_j traverses the edge $\hat{e}_h=(v_s,v_t)\in\hat{\mathcal{E}}$ and serves node v_t, 0 otherwise
$\hat{y}_{p,q,j}$	Decision variable for the sub-optimal problem which is 1 if the consecutive edges $(\hat{e}_p,\hat{e}_q)\in\hat{\mathcal{E}}_c$ are traversed by robot r_j, 0 otherwise

III. PRELIMINARIES

We model the orchard field by means of a directed graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$, referred to as *field* graph, with set of vertices (or nodes) $\mathcal{V}=\{v_0,v_1,\ldots,v_n\}$ and set of edges (or arcs) $\mathcal{E}\subseteq\{(v_i,v_j)\,|\,v_i,v_j\in\mathcal{V},i\neq j\}$. Each vertex v_i is associated with a possible location where an agricultural task can be carried out and we denote its position in the field with respect to a common frame Σ_w by $p_i\in\mathbb{R}^3$. As instance, a location close to each plant can be selected for tasks involving harvesting, pruning or applications of herbicide, pesticide or fertilizer. An edge (v_i,v_j) is associated with a possible passage, e.g., line-of-sight, from location v_i to v_j for which we make the following assumption.

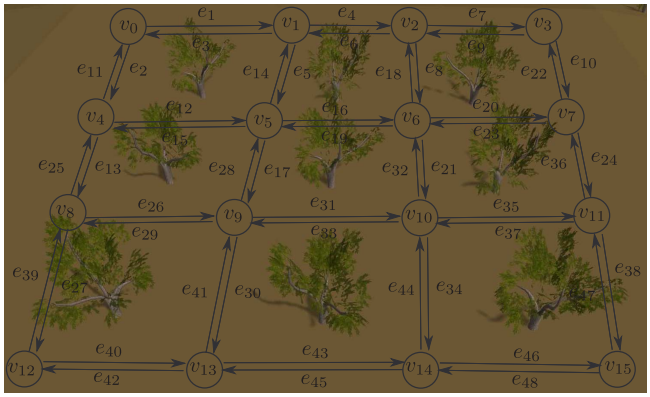


Fig. 2. Example of field graph associated with a field. Nodes are represented by circles and edges by arrows.

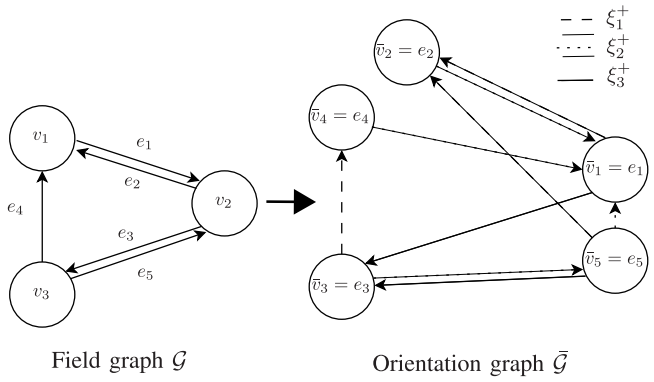


Fig. 3. Example of field graph (left) and respective orientation graph (right). The sets of edges $\xi_1^+, \xi_2^+, \xi_3^+$ are marked with dashed, dotted and solid lines, respectively.

Assumption 1: The physical passages in the field, associated with edges in the field graph, do not require maneuvering operations.

The above assumption implies that the physical passages can be approximated as straight paths. This is only assumed to simplify the mathematical analysis in the following. In case non-straight curves need to be associated with the edges, the additional information about the orientation to enter and exit each node as well as possible turns in the path should be integrated within the formulation.

Without loss of generality, we order the set of edges and denote the kth one by $e_k \in \mathcal{E}$. As instance, we sort the edges in increasing order with respect to the left vertex of the pair. An example of field graph associated with a given field is provided in Figure 2. Nodes are represented by circles and edges by arrows. Each edge $e_k = (v_i, v_j)$ is labeled with its physical length $l_k = \|p_i - p_j\|$, where $\|\cdot\|$ denotes the Euclidean norm. The graph is defined complete if all the possible edges (v_i, v_j) $\forall v_i, v_j \in \mathcal{V}$ with $v_i \neq v_j$ exist, i.e., no self-loops are allowed. We introduce the following sets:

$$\delta_i^+ = \{(v_j, v_i) \in \mathcal{E} \mid v_j \in \mathcal{V}\}$$
$$\delta_i^- = \{(v_i, v_j) \in \mathcal{E} \mid v_j \in \mathcal{V}\}$$

where δ_i^+ is the set of edges having v_i as head, while δ_i^- is the set of edges having v_i as tail. Based on these, the set $\mathcal{E}_c$ of all the pairs of consecutive edges in the graph is defined:

$$\mathcal{E}_c = \{(e_p, e_q) \mid e_p \in \delta_i^+, e_q \in \delta_i^-, v_i \in \mathcal{V}\} \subseteq \mathcal{E} \times \mathcal{E}.$$

Given a pair of consecutive edges $(e_p, e_q) \in \mathcal{E}_c$, where $e_p = (v_t, v_v)$ and $e_q = (v_v, v_h)$, with $\{v_t, v_v, v_h\} \in \mathcal{V}$, the vectors $u_p = p_t - p_v$ and $u_q = p_h - p_v$ are introduced. Based on these, the angle $\varphi_{p,q}$ between e_p and e_q, in the plane containing the respective segments, is computed as follows:

$$\varphi_{p,q} = \arccos\left(\frac{u_p^T u_q}{\|u_p\|\|u_q\|}\right). \tag{1}$$

The maximum value of $\varphi_{p,q}$ is equal to π in case the edges e_p and e_q are parallel and with opposite directions, meaning that the robot has to completely turn back along its path to traverse e_p and e_q, while the minimum value is equal to 0 in case the edges are parallel and with same directions, meaning that the robot does not have to perform any steering to cross the edges. We consider that a depot location is present in the field which, without loss of generality, is associated with the vertex $v_0 \in \mathcal{V}$. Moreover, we consider that a subset $\mathcal{V}_s \subseteq \mathcal{V} \setminus \{v_0\}$ of vertices must be served, i.e., agricultural tasks must be carried out only in the respective locations of the field. This set excludes the depot vertex v_0. Note that large-scale orchards are generally characterized by a high number of nodes, i.e., $|\mathcal{V}| \gg 1$ with $|\cdot|$ denoting the cardinality of the set $(\cdot)$, where only few of them must be processed with agricultural tasks, i.e., it generally holds $|\mathcal{V}_s| \ll |\mathcal{V}|$.

Given the field graph, we build an additional graph $\bar{\mathcal{G}} = (\bar{\mathcal{V}}, \bar{\mathcal{E}})$, called *orientation* graph which models the relations between couples of edges in $\mathcal{E}$. These relations are needed to handle turning costs with generic field topologies. In detail, each vertex $\bar{v}_k \in \bar{\mathcal{V}}$ is associated with an edge e_k of the field graph, i.e, $\bar{v}_k = e_k = (v_t, v_i)$ $\forall e_k \in \mathcal{E}$, and each edge $\bar{e}_k \in \bar{\mathcal{E}}$ is associated with consecutive edges of the field graph, i.e., $\bar{e}_k = (\bar{v}_p, \bar{v}_q)$ exists if $(e_p, e_q) \in \mathcal{E}_c$. An edge $\bar{e}_k = (\bar{v}_p, \bar{v}_q)$ is labeled with a tuple $(\bar{l}_k, \varphi_{p,q})$, where $\bar{l}_k = l_p + l_q$ is the overall length of the edges e_p and e_q and $\varphi_{p,q}$ is the angle between the edges computed according to (1). Moreover, we introduce the following sets:

$$\bar{\xi}_i^+ = \{\bar{e}_k = (\bar{v}_p, \bar{v}_q) \mid \bar{v}_q \in \delta_i^+, \bar{e}_k \in \bar{\mathcal{E}}\} \tag{2}$$
$$\bar{\xi}_i^- = \{\bar{e}_k = (\bar{v}_p, \bar{v}_q) \mid \bar{v}_p \in \delta_i^-, \bar{e}_k \in \bar{\mathcal{E}}\} \tag{3}$$

where $\bar{\xi}_i^+$ is the set of edges in $\bar{\mathcal{E}}$ which eventually flow into node $v_i \in \mathcal{V}$ of the field graph, and $\bar{\xi}_i^-$ is the set of edges in $\bar{\mathcal{E}}$ which originate from node $v_i \in \mathcal{V}$ of the field graph. Figure 3 reports another illustrative example of a field graph $\mathcal{G}$ (left), composed of three nodes and five edges and respective orientation graph $\bar{\mathcal{G}}$ (right). The sets of edges $\xi_1^+, \xi_2^+, \xi_3^+$ are highlighted with dashed, dotted and solid lines, respectively.

A set $\mathcal{R} = \{r_1, \ldots, r_m\}$ of m robots is available to perform agricultural tasks in the field. As it occurs in real-world contexts, we consider that each robot r_j has a limited energy capacity denoted by K_j. Moreover, for each robot r_j, the following temporal costs are defined:

- $c_j^e \in \mathbb{R}$: unit temporal cost for traversing an edge, i.e., average amount of seconds to travel one meter along an edge;

- $c_j^t \in \mathbb{R}$: unit temporal cost for turning, i.e., average amount of seconds needed to make a turn of one radian;
- $c_{i,j}^s \in \mathbb{R}$, $\forall v_i \in \mathcal{V}_s$: temporal service cost for performing an agricultural task on node v_i. Note that the dependency on the node v_i is required as the different agricultural tasks may require different times to be executed.

Similarly, we introduce the respective unit energy costs for each robot, representing the consumed energy in the time unit. These unit energy costs for traversing an edge, turning and serving a node by robot r_j are denoted by ϵ_j^e, ϵ_j^t, ϵ_j^s, respectively. We make the following assumptions.

Assumption 2: Robots are equipped with a navigation controller that enables traveling through the assigned routes.

Assumption 3: All the robots depart from a depot station v_0 and must return to it.

Assumption 2 can be easily satisfied by resorting to existing navigation architectures, e.g., [9], [50], which are out of the scope of this paper, while Assumption 3 is realistic, for example, in typical PA settings where all the robots are usually stored in a warehouse. Table II summarizes the main notations of the paper.

IV. Multi-Steiner TSP

In this section, we define the main problem addressed in this paper and describe the proposed formulation to solve it.

A. Problem Statement

Let $\mathcal{B}$ be the binary set, i.e., $\mathcal{B} = \{0, 1\}$. We introduce the following binary decision variables: $x_{k,j} \in \mathcal{B}$, $\forall \bar{e}_k \in \bar{\mathcal{E}}, r_j \in \mathcal{R}$, encoding the route assigned to robot r_j, which is 1 if robot r_j has to traverse the edge $\bar{e}_k$ of the orientation graph, 0 otherwise, and $s_{i,j} \in \mathcal{B}, \forall v_i \in \mathcal{V}_s, r_j \in \mathcal{R}$, encoding the nodes to serve by robot r_j along the assigned route, which is 1 if robot r_j serves the vertex v_i belonging to $\mathcal{V}_s$, i.e., it performs an agricultural task on node $v_i \in \mathcal{V}_s$, 0 otherwise. Note that the condition $x_{k,j} = 1$, with $\bar{e}_k = (\bar{v}_p, \bar{v}_q)$ implies that robot r_j has to traverse both edges $e_p \in \mathcal{E}$ and $e_q \in \mathcal{E}$ of the field graph. Finally, we define the aggregate decision vectors $x_j = [x_{1,j} \dots x_{|\bar{\mathcal{E}}|,j}]^T \in \mathcal{B}^{|\bar{\mathcal{E}}|}$ and $s_j = [s_{1,j} \dots s_{|\mathcal{V}_s|,j}]^T \in \mathcal{B}^{|\mathcal{V}_s|}$ collecting the variables $x_{k,j}$ $\forall \bar{e}_k \in \bar{\mathcal{E}}$ and $s_{i,j}$ $\forall v_i \in \mathcal{V}_s$ for robot r_j, respectively.

Problem 1: Consider a set $\mathcal{R}$ of (possibly) heterogeneous robots with energy capacity K_j, $j \in \mathcal{R}$. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the field graph, with orientation graph $\bar{\mathcal{G}} = (\bar{\mathcal{V}}, \bar{\mathcal{E}})$, and $\mathcal{V}_s \subset \mathcal{V}$ be the subset of nodes where agricultural tasks must be carried out. We aim to optimally determine for each robot r_j:

i) which agricultural tasks it has to perform, i.e., $s_{i,j}$, $\forall v_i \in \mathcal{V}_s$,

ii) its route, i.e., $x_{k,j}$, $\forall \bar{e}_k \in \bar{\mathcal{E}}$,

while complying with energy capacity constraints. The optimality is intended to minimize the maximum temporal cost as well as the cumulative temporal costs by all the robots, comprising costs to traverse the edges, to turn and to perform agricultural tasks.

As discussed in the following, the optimality notion in Problem 1 provides a good compromise between balancing the workload among the robots, within the energy constraints, and minimizing the overall execution time.

B. Problem Formulation

We are now ready to present the proposed multi-Steiner TSP formulation for solving Problem 1. Let us introduce the following aggregate temporal costs for each robot r_j:

$$C_j^e(x_j) = \sum_{\bar{e}_k \in \bar{\mathcal{E}}} c_j^e \cdot \bar{l}_k \cdot x_{k,j} \tag{4}$$

$$C_j^t(x_j) = \sum_{\bar{e}_k = (\bar{v}_p, \bar{v}_q) \in \bar{\mathcal{E}}} c_j^t \cdot \varphi_{p,q} \cdot x_{k,j} \tag{5}$$

$$C_j^s(s_j) = \sum_{v_i \in \mathcal{V}_s} c_{i,j}^s \cdot s_{i,j} \tag{6}$$

where C_j^e in (4) represents the overall temporal cost of robot r_j to traverse the edges, C_j^t in (5) is the overall temporal cost of robot r_j to turn, and C_j^s in (6) is the overall service time for agricultural tasks performed by robot r_j. Note that the turning costs are easily computed in (5) thanks to the orientation graph model which allows to preserve the information of which consecutive edges are crossed and the respective turn to transit from one to the next. We define the overall temporal cost of robot j as follows

$$C_j(x_j, s_j) = \alpha\, C_j^e(x_j) + \beta\, C_j^t(x_j) + \gamma\, C_j^s(s_j) \tag{7}$$

with $\alpha, \beta, \gamma \in \mathbb{R}^+$ positive weights, and denote the maximum cost among the robots by

$$C_{\max} = \max_{r_j \in \mathcal{R}} C_j(x_j, s_j). \tag{8}$$

As common in multi-objective optimization problems [51] with weighted sum, the weights α, β, γ in (7) influence the priority assigned in minimizing the temporal cost of traversing edges, turning, and serving nodes in the solution, respectively, i.e., the higher a weight compared to the others, the more the optimal solution will aim to minimize the respective cost.

At this point, we can formally state our novel multi-Steiner TSP formulation which is expressed as an Integer Linear Programming (ILP) problem:

$$\min_{\substack{x_j,\, s_j \\ \forall r_j \in \mathcal{R}}} \quad C_{\max} + \sum_{r_j \in \mathcal{R}} C_j(x_j, s_j) \tag{9a}$$

$$\text{s.t.} \sum_{r_j \in \mathcal{R}} s_{i,j} = 1, \quad \forall v_i \in \mathcal{V}_s \tag{9b}$$

$$\sum_{\bar{e}_p \in \xi_i^+} x_{p,j} \geq s_{i,j}, \ \forall v_i \in \mathcal{V}_s, \forall r_j \in \mathcal{R} \tag{9c}$$

$$\sum_{\bar{e}_q \in \xi_0^-} x_{q,j} \geq s_{i,j}, \ \forall v_i \in \mathcal{V}_s, \forall r_j \in \mathcal{R} \tag{9d}$$

$$\sum_{\bar{e}_p \in \xi_i^+} x_{p,j} = \sum_{\bar{e}_q \in \xi_i^-} x_{q,j}, \ \forall v_i \in \mathcal{V}, \forall r_j \in \mathcal{R} \tag{9e}$$

$$\sum_{\bar{e}_p \in \xi_i^-} f_{p,j} - \sum_{\bar{e}_q \in \xi_i^+} f_{q,j} = s_{i,j}, \quad \forall v_i \in \mathcal{V}_s, \forall r_j \in \mathcal{R} \tag{9f}$$

$$\sum_{\bar{e}_p \in \xi_i^-} f_{p,j} = \sum_{\bar{e}_q \in \xi_i^+} f_{q,j}, \ \forall v_i \in \mathcal{V} \setminus \{\mathcal{V}_s\}, v_i \neq v_0, \forall r_j \in \mathcal{R} \tag{9g}$$

$$\epsilon_j^e C_j^e(x_j) + \epsilon_j^t C_j^t(x_j) + \epsilon_j^s C_j^s(s_j) \leq K_j, \quad \forall r_j \in \mathcal{R} \tag{9h}$$

$$s_{i,j} \in \mathcal{B}, \quad \forall v_i \in \mathcal{V}_s, \ \forall r_j \in \mathcal{R} \tag{9i}$$

$$x_{k,j} \in \mathcal{B}, \quad \forall \bar{e}_k \in \bar{\mathcal{E}}, \ \forall r_j \in \mathcal{R} \tag{9j}$$

$$0 \leq f_{k,j} \leq |\mathcal{V}_s| \cdot x_{k,j}, \ \forall \bar{e}_k \in \bar{\mathcal{E}}, \ \forall r_j \in \mathcal{R}. \tag{9k}$$

The objective, according to (9a), is to determine the decision vectors $x_j, s_j, \forall r_j \in \mathcal{R}$ while minimizing the maximum temporal cost $C_{\max}$ and the cumulative weighted temporal cost of the robotic platforms, i.e., $\sum_{r_j \in \mathcal{R}} C_j(x_j, s_j)$. In this way, we aim to minimize the total time to perform all the agricultural tasks, while avoiding unnecessary motions by all the robots. Indeed, in the case only $C_{\max}$ is considered in the objective function, the other robots $r_j \in \mathcal{R}$ are allowed to arbitrarily move within the constraints as long as $C_j < C_{\max}$ even if these motions are not aimed to serve nodes. On the other hand, if only the cumulative temporal cost $\sum_{r_j \in \mathcal{R}} C_j$ is considered in the objective function, a solution may allocate most of the work to one robot without balancing the load and leading to a higher $C_{\max}$. Note that practical benefits result from encouraging load-balancing. In particular, the more balanced the load, the more uniform the robots' wear and tear in the long run and, therefore, the longer their lifetime. Moreover, considering the typical presence of uncertainties during execution in terms of times and energy, such as in [52], work distribution provides increased robustness if these measures are underestimated at allocation time. This motivates why a combined objective function is considered. However, in case one only wants to optimize the time efficiency, the proposed formulation can be adapted to the purpose in a straightforward manner by excluding the term $\sum_{r_j \in \mathcal{R}} C_j(x_j, s_j)$ in the cost function in (9a).

As far as the constraints are concerned, the following are defined:

- Equation (9b) implies that each node $v_i \in \mathcal{V}_s$ is served by exactly one robot. However, this constraint does not prevent other robots or the same serving robot from *transiting* on node v_i multiple times if needed.
- Equation (9c) states that, in order for a robot to serve a node $v_i \in \mathcal{V}_s$ and thus perform agricultural tasks on it, the node must be entered by the robot, i.e., if $s_{i,j} = 1$ then the robot v_j must traverse an edge $\bar{e}_p = (\bar{v}_s, \bar{v}_t) \in \xi_i^+$ which eventually enters the node.
- Equation (9d) implies that, if a robot r_j has to serve at least one node, i.e., it exists $s_{i,j} = 1$ for $v_i \in \mathcal{V}_s$, then the robot has to exit the depot v_0. In this way, if a robot is not supposed to carry out any agricultural tasks, no unnecessary motion is performed.
- Equation (9e) states that each time a robot r_j enters a node v_i, i.e., it exists $x_{p,j} = 1$ for $\bar{e}_p \in \xi_i^+$, the same robot must also exit it, i.e. it must hold $x_{q,j} = 1$ for $\bar{e}_q \in \xi_i^-$.
- Equations (9f)-(9g) refer to the single commodity flow formulation [53] which allows avoiding the generation of disjoint loops in the robots' routes. In detail, each robot virtually transports commodities through edges: a commodity is released when the robot serves a node, while no commodity change occurs when the robot only *transits* on a node. Equation (9f) imposes the flow adaptation if a node $v_i \in \mathcal{V}_s$, which can be *potentially* served, is traversed by robot r_j: if the node is served, i.e., if it holds $s_{i,j} = 1$, then the amount of commodities is reduced by one unit, otherwise no commodities are released. Equation (9g) imposes the flow conservation on transit nodes, i.e., nodes that should not be served. When these nodes are transited, the number of commodities must be unaffected. With this formulation, even if a node is served by a certain robot, the remaining ones are still allowed to pass through it, without modifying their flow.
- Equation (9h) bounds the overall energy consumption of a robot r_j to its capacity K_j. The energy costs for edge traversing, turning and serving operations are taken into account.
- Equations (9i)-(9j) impose the binary nature of the decision variables $x_{k,j}, s_{i,j} \ \forall i, j, k$;
- Equation (9k) regulates the commodity flow variables $f_{k,j}$, imposing that, for each edge $\bar{e}_k$ and for each robot r_j, the amount of commodities $f_{k,j}$ passing through the edge *i)* does not exceed the number of nodes to be visited $|\mathcal{V}_s|$ if the edge is traversed, and *ii)* is equal to 0 otherwise. This guarantees that an edge is present in the robot's route if a commodity is passed through this edge by the same robot.

Notably, the proposed formulation allows to easily specify different locations where agricultural tasks should be carried out by a heterogeneous team of robots, while minimizing the maximum and the cumulative temporal costs. Moreover, the formulation is completely flexible in regards to the field topology, allowing turning costs to be included in any case.

Concerning the problem complexity, it can be proven that the formulated multi-Steiner TSP is NP-hard, i.e., it cannot be solved by a polynomial algorithm if P $\neq$ NP. More specifically, by recalling that the TSP is NP-hard [54] and is a special case of the single Steiner TSP in which all the nodes must be served, i.e., $\mathcal{V}_s \equiv \mathcal{V}$, the NP-hardness of the single Steiner TSP follows [54]. This NP-hardness is thus inherited also by the multiple robots case addressed in this paper [46]. Notably, heuristics like in [35] and [40] could be designed to cope with the NP-hardness of the problem and make the solution computation more affordable. However, this is beyond the scope of this paper, which focuses on the problem *formulation* aspects to perform agricultural tasks by multiple robots in realistic PA settings, rather than the algorithmic aspects for efficiently solving the problem. Nevertheless, in the following we propose a sub-optimal formulation which reduces the problem dimensionality thus mitigating its computational load, as validated in Section VI.

V. Sub-Optimal Formulation

In order to properly account for the turning costs, the entire orientation graph must be considered as in (9a) and all the maneuvering costs associated with turns must be counted. However, in case the number of agricultural tasks to perform is significantly lower than the number of possible field locations, i.e., $|\mathcal{V}_s| \ll |\mathcal{V}|$, as typical for PA settings, a sub-optimal formulation for addressing Problem 1 which relaxes the turning costs at the service nodes only may be

convenient. Algorithm 1 summarizes the two-phase procedure to set-up and solve the sub-optimal problem starting from the orientation graph $\bar{\mathcal{G}}$. The basic idea is to decompose the original formulation into multiple polynomial problems which require overall substantially less computational time to be solved. This enables involving a reduced number of variables in the final optimization problem at the cost of sub-optimality, for which we characterize a bound in the following.

Algorithm 1 Procedure to Solve Sub-Optimal Formulation

Require: Service nodes $\mathcal{V}_s$, Orientation graph $\bar{\mathcal{G}}$, coefficients c_j^k, $c_{i,j}^s$ with $k = e, t$, $\forall v_i \in \mathcal{V}_s$, ϵ_j^k with $k = e, t, s$ $\forall r_j \in \mathcal{R}$

Phase 1 - Build service graph $\hat{\mathcal{G}}$

1: $\hat{\mathcal{V}} \leftarrow \mathcal{V}_s \cup \{v_0\}$
2: $\hat{\mathcal{E}} \leftarrow \emptyset$
3: **for each** $v_s, v_t \in \hat{\mathcal{V}}$ **do**
4: $\hat{e}_h := (v_s, v_t)$
5: **for each** $r_j \in \mathcal{R}$ **do**
6: $\hat{L}_{h,j} \leftarrow$ local optimum$(\bar{\mathcal{G}}, v_s, v_t, c_j^k)$ [eq. (10)]
7: **end for**
8: $\hat{\mathcal{E}} \leftarrow \hat{\mathcal{E}} \cup \{\hat{e}_h$with label $\hat{L}_h\}$
9: **end for**

Phase 2 - Find sub-optimal solution in $\hat{\mathcal{G}}$

1: $\hat{x}_{h,j} \leftarrow$global sub-optimum$(\hat{\mathcal{G}}, c_j^k, c_{i,j}^s, \epsilon_j^k)$ [eq. (15)]

As first step, we build an additional graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$ (Phase 1 of Algorithm 1), called *service* graph, which embeds the information of minimum cost paths between service nodes. The set of vertices $\hat{\mathcal{V}}$ only comprises the nodes to serve $\mathcal{V}_s$ and the depot v_0, i.e., $\hat{\mathcal{V}} = \mathcal{V}_s \cup \{v_0\}$ (line 1.1). Concerning the edges, a complete graph is built, i.e., $\hat{\mathcal{E}} = \hat{\mathcal{V}} \times \hat{\mathcal{V}}$ (lines 1.2-1.9), in which each edge $\hat{e}_h = (v_s, v_t) \in \hat{\mathcal{E}}$ represents the paths with minimum cost from the start node v_s to the target one v_t for each robot. In detail, a m-dimensional cost vector $\hat{L}_h = [\hat{L}_{h,1}, .., \hat{L}_{h,m}]$ is associated with edge $\hat{e}_h$ whose jth component is a tuple $\hat{L}_{h,j} = (\hat{L}_{h,j}^e, \hat{L}_{h,j}^t)$ (line 1.6), where $\hat{L}_{h,j}^e$ is the overall temporal cost for robot r_j to traverse the edges from v_s to v_t, while $\hat{L}_{h,j}^t$ is the overall temporal cost for robot r_j for turning operations at the transit nodes. Additionally, the sets $\hat{\delta}_i^+$,$\hat{\delta}_i^-$ are defined for each node $v_i \in \hat{\mathcal{V}}$, which, similarly to the sets δ_i^+, δ_i^- of the field graph, are composed of the edges in $\hat{\mathcal{E}}$ entering and exiting vertex v_i, respectively. Furthermore, the set $\hat{\mathcal{E}}_c$ of consecutive edges in $\hat{\mathcal{E}}$ is defined.

To optimally define the cost vector $\hat{L}_h$ for each edge $\hat{e}_h = (v_s, v_t) \in \hat{\mathcal{E}}$, we resort to the orientation graph $\bar{\mathcal{G}}$. Let us introduce the decision variable $a_{k,h,j}$ which is 1 if the edge $\bar{e}_k \in \bar{\mathcal{E}}$ belongs to the path from v_s to v_t with minimum cost for robot r_j, 0 otherwise, and the respective collective vector $a_{h,j}$ obtained by considering all $\bar{e}_k \in \bar{\mathcal{E}}$. The following optimization problem is solved $\forall \hat{e}_h = (v_s, v_t) \in \hat{\mathcal{E}}, r_j \in \mathcal{R}$:

$$\min_{a_{h,j}} \ \alpha\, C_j^e(a_{h,j}) + \beta\, C_j^t(a_{h,j}) \tag{10a}$$

$$\text{s.t.} \sum_{\bar{e}_p \in \xi_i^+} a_{p,h,j} - \sum_{\bar{e}_q \in \xi_i^-} a_{q,h,j} = \begin{cases} 1 & \text{if } v_i = v_s \\ -1 & \text{if } v_i = v_t \\ 0 & \text{otherwise,} \end{cases} \quad \forall v_i \in \mathcal{V} \tag{10b}$$

$$a_{k,h,j} \in \mathcal{B}, \quad \forall \bar{e}_k \in \bar{\mathcal{E}} \tag{10c}$$

where α and β are selected as in (9a). Based on the solution $a_{h,j}$ to the above problem, it holds $\hat{L}_{h,j}^e = C_j^e(a_{h,j})$, $\hat{L}_{h,j}^t = C_j^t(a_{h,j})$. The optimization problem in (15) is formulated as a shortest path problem [55] which can be thus solved through polynomial algorithms [55]. More specifically, according to the objective function (10a), it minimizes the overall temporal cost for robot r_j to traverse edges and to execute the respective turnings from v_s to v_t. As far as the constraints are concerned, equation (10b) regulates the value of the decision variable $a_{k,h,j}$ imposing that exactly one edge is selected to exit the start node v_s and one is selected to enter the target node v_t, as well as that all the other edges comprising the optimal path are selected. Finally, equation (10c) defines the binary nature of the decision variables $a_{k,h,j}$.

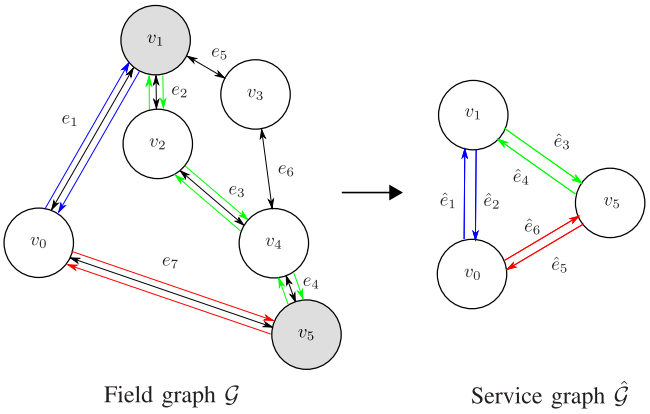


Fig. 4. Example of field graph (left) and respective service graph (right). Nodes v_1 and v_5, highlighted in gray, are required to be visited. The shortest paths between pairs of nodes v_1 and v_5 and the depot node v_0 are highlighted with colored arrows in the field graph.

We denote the path in the field graph $\mathcal{G}$ associated with the solution $a_{h,j}$ by $\mathcal{P}_{h,j} = \{e_{h,j}^{[1]}, \ldots, e_{h,j}^{[\omega]}\}$, where $e_{h,j}^{[i]} \in \mathcal{E}$ denotes the ith edge of the path $\mathcal{P}_{h,j}$ with cardinality ω. To transition from the orientation graph to the field graph and thus define the path $\mathcal{P}_{h,j}$, we consider, as in Section IV, that, if $a_{k,h,j} = 1$, with $\bar{e}_k = (\bar{v}_p, \bar{v}_q)$, then both edges $e_p \in \mathcal{E}$ and $e_q \in \mathcal{E}$ belong to the path. Note that different optimal paths from v_s to v_t can be found for the different robots $r_j \in \mathcal{R}$ due to their heterogeneity, i.e., to their different unit costs for turning and for traversing an edge.

Remark 1: In the case of homogeneous robots, the costs c_j^e, c_j^t are coincident for each robot $r_j \in \mathcal{R}$, therefore a unique path for all the robots is found that connects v_s and v_t. This implies that problem (10) can be solved only once for each edge $\hat{e}_h \in \hat{\mathcal{E}}$ and a single label $\hat{L}_h$ is generated.

Figure 4 depicts an example of service graph (on the right) obtained from a field graph (on the left). In the example, homogeneous robots are assumed and two nodes (highlighted

in gray) are required to be visited, namely $\mathcal{V}_s = \{v_1, v_5\}$. The service graph is composed of the set of nodes $\hat{\mathcal{V}} = \{v_0, v_1, v_5\}$ and the set of edges $\hat{\mathcal{E}} = \{\hat{e}_1, \ldots, \hat{e}_6\}$ obtained by the shortest paths connecting the respective origin and destination nodes. Colored arrows in the figure are used to denote the shortest paths used in the service graph. For example, edges $\hat{e}_1 = (v_0, v_1)$ and $\hat{e}_2 = (v_1, v_0)$ correspond to the shortest paths $\mathcal{P}_{1,j} = \{e_1\}$ and $\mathcal{P}_{2,j} = \{e_1\}$ (blue arrows) from nodes v_0 to v_1 and v_1 to v_0, respectively, for a generic robot r_j, while edges $\hat{e}_3 = (v_1, v_5)$ and $\hat{e}_4 = (v_5, v_1)$ correspond to the shortest paths $\mathcal{P}_{3,j} = \{e_2, e_3, e_4\}$ and $\mathcal{P}_{4,j} = \{e_4, e_3, e_2\}$ (green arrows) from nodes v_1 to v_5 and v_5 to v_1, respectively.

At this point, we are ready to exploit the service graph to define a sub-optimal solution to (9a) (Phase 2 of Algorithm 1). Let us introduce the binary decision variable $\hat{x}_{k,j}$ $\forall \hat{e}_k \in \hat{\mathcal{E}}$, $r_j \in \mathcal{R}$ which is 1 when robot r_j traverses the edge $\hat{e}_h = (v_s, v_t)$ and serves node v_t, and is 0 otherwise, and the variable $\hat{y}_{p,q,j}$ $\forall (\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c$, $r_j \in \mathcal{R}$ which is 1 if the consecutive edges $(\hat{e}_p, \hat{e}_q)$ are traversed by robot r_j, 0 otherwise. Note that no service is made if $v_t = v_0$. Moreover, we define the angle between two consecutive edges of the service graph $\hat{e}_p, \hat{e}_q \in \hat{\mathcal{E}}_c$ traversed by robot r_j as follows

$$\hat{\varphi}_{p,q,j} = \varphi_{[\omega]_{p,j},[1]_{q,j}} \tag{11}$$

meaning that it represents the angle between the last edge $(e^{[\omega]}_{p,j})$ of the path $\mathcal{P}_{p,j}$ associated with $\hat{e}_p$ and the first edge $(e^{[1]}_{q,j})$ of the path $\mathcal{P}_{q,j}$ associated with $\hat{e}_q$. Note that the angle $\hat{\varphi}_{p,q,j}$ depends on the robot r_j since, as mentioned above, different paths $\mathcal{P}_{h,j}$ may be associated with different robots r_j. Similarly to eqs. (4)-(6), the following costs are introduced

$$\hat{C}^e_j = \sum_{\hat{e}_k \in \hat{\mathcal{E}}} \hat{L}^e_{k,j} \cdot \hat{x}_{k,j} \tag{12}$$

$$\hat{C}^t_j = \hat{C}^{t_t}_j + \hat{C}^{t_s}_j \quad \text{with} \quad \begin{cases} \hat{C}^{t_t}_j = \displaystyle\sum_{\hat{e}_k \in \hat{\mathcal{E}}} \hat{L}^t_{k,j} \cdot \hat{x}_{k,j} \\ \hat{C}^{t_s}_j = \displaystyle\sum_{(\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c} c^t_j \cdot \hat{\varphi}_{p,q,j} \cdot \hat{y}_{p,q,j} \end{cases} \tag{13}$$

$$\hat{C}^s_j = \sum_{v_i \in \mathcal{V}_s} \sum_{\hat{e}_p \in \hat{\delta}^+_i} c^s_{i,j} \cdot \hat{x}_{p,j} \tag{14}$$

where $\hat{C}^e_j$ in (12) is the overall temporal cost of robot r_j to traverse the edges, $\hat{C}^t_j$ in (13) is the one for turning made up of the component $\hat{C}^{t_t}_j$, related to transit nodes, and the component $\hat{C}^{t_s}_j$, related to nodes to serve, and C^s_j in (14) is the overall service time of robot r_j. Moreover, we introduce the following aggregate cost for robot r_j

$$\hat{C}_j = \alpha\, \hat{C}^e_j + \beta\, \hat{C}^t_j + \gamma\, \hat{C}^s_j$$

with α, β and γ as in (9a), whose maximum value is denoted by

$$\hat{C}_{\max} = \max_{r_j \in \mathcal{R}} \hat{C}_j.$$

The following optimization problem is formulated on the basis of the service graph $\hat{\mathcal{G}}$ (Phase 2 in Algorithm 1, line 2.1):

$$\min_{\substack{\hat{x}_{k,j}, \hat{y}_{p,q,j} \\ \forall k, j, p, q}} \quad \hat{C}_{\max} + \sum_{r_j \in \mathcal{R}} \hat{C}_j \tag{15a}$$

$$\text{s.t.} \sum_{r_j \in \mathcal{R}} \sum_{\hat{e}_p \in \hat{\delta}^+_i} \hat{x}_{p,j} = 1, \quad \forall v_i \in \mathcal{V}_s \tag{15b}$$

$$\sum_{r_j \in \mathcal{R}} \sum_{\hat{e}_q \in \hat{\delta}^-_i} \hat{x}_{q,j} = 1, \quad \forall v_i \in \mathcal{V}_s \tag{15c}$$

$$\sum_{r_j \in \mathcal{R}} \sum_{\hat{e}_p \in \hat{\delta}^-_i} \hat{x}_{p,j} = \sum_{r_j \in \mathcal{R}} \sum_{\hat{e}_q \in \hat{\delta}^+_i} \hat{x}_{q,j}, \quad \forall v_i \in \hat{\mathcal{V}} \tag{15d}$$

$$\sum_{\substack{\hat{e}_k \in \hat{\mathcal{E}}: \\ \hat{e}_k = (v_s, v_t), \\ v_s, v_t \in \mathcal{S}}} \hat{x}_{k,j} \leq |\mathcal{S}| - 1, \quad \forall \mathcal{S} \subset \hat{\mathcal{V}}, \mathcal{S} \neq \emptyset \tag{15e}$$

$$\hat{x}_{p,j} + \hat{x}_{q,j} \leq 1 + \hat{y}_{p,q,j}, \quad \forall (\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c \tag{15f}$$

$$\epsilon^e_j \hat{C}^e_j + \epsilon^t_j \hat{C}^t_j + \epsilon^s_j \hat{C}^s_j \leq K_j, \quad \forall r_j \in \mathcal{R} \tag{15g}$$

$$\hat{x}_{k,j} \in \mathcal{B}, \quad \forall \hat{e}_k \in \hat{\mathcal{E}}, \ \forall r_j \in \mathcal{R} \tag{15h}$$

$$\hat{y}_{p,q,j} \in \mathcal{B}, \quad \forall (\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c, \ \forall r_j \in \mathcal{R}. \tag{15i}$$

By analogy with the formulation in Section IV, we here aim to minimize in (15a) the maximum weighted temporal cost $\hat{C}_{\max}$ and the cumulative weighted temporal costs of all the robotic platforms, resorting to the *service* graph. The following constraints are defined:

- Equations (15b)-(15c) require that each node to serve $v_i \in \mathcal{V}_s$ is entered and exited by exactly one robot.
- Equation (15d) states that each time a robot enters a node $v_i \in \hat{\mathcal{V}}$, the robot must also exit the same node.
- Equation (15e) allows avoiding sub-tours [56]. More specifically, for each non-empty subset $\mathcal{S}$ of $\hat{\mathcal{V}}$, the inequality in (15e) forces the selection of a number of edges with vertices in $\mathcal{S}$ lower than $|\mathcal{S}|$, thus preventing the creation of sub-tours of length $|\mathcal{S}|$.
- Equation (15f) determines the behavior of the variables $\hat{y}_{p,q,j}$ $\forall (\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c, \forall r_j \in \mathcal{R}$ needed for the turning costs at the service nodes $\hat{C}^{t_s}_j$ in (13). More specifically, if a robot r_j traverses the paths associated with the consecutive edges $\hat{e}_p$ and $\hat{e}_q$, i.e., $\hat{x}_{p,j} = \hat{x}_{q,j} = 1$, the inequality forces the variable $\hat{y}_{p,q,j}$ to be greater than or equal to 1. On the contrary, if the consecutive edges are not both assigned to robot r_j, the variable $y_{p,q,j}$ can be equal to 0. Note that, as we minimize the turning costs, these variables will be equal to 0 in the latter case.
- Equation (15g), similar to (9h), bounds the overall energy consumption of robot r_j to its capacity K_j.
- Equations (15h)-(15i) impose the binary nature of the variables $\hat{x}_{i,j}$ and $\hat{y}_{p,q,j}$, $\forall i, j, p, q$.

Based on the solution to (15), the nodes that each robot r_j has to serve are determined by the variables $\hat{x}_{h,j}$, i.e., $s_{i,j} = 1$ if edge $\hat{e}_h$ terminates in v_i, i.e., $\hat{e}_h = (v_s, v_i)$, with $v_i \neq v_0$ and it is assigned to robot r_j, i.e., $\hat{x}_{h,j} = 1$, $s_{i,j} = 0$ otherwise. Concerning the route assigned to each robot r_j, it is determined by the shortest paths associated with the variables $\hat{x}_{h,j} = 1$, i.e., $x_{k,j} = 1$ with $e_k \in \mathcal{E}$ if $\hat{x}_{h,j} = 1$ and $a_{k,h,j} = 1$, $x_{k,j} = 0$ otherwise.

Concerning the problem complexity, it inherits the NP-hardness of the orientation graph-based formulation in (9a), where the P problems for finding the shortest paths in (10) are negligible if P $\neq$ NP. However, the number

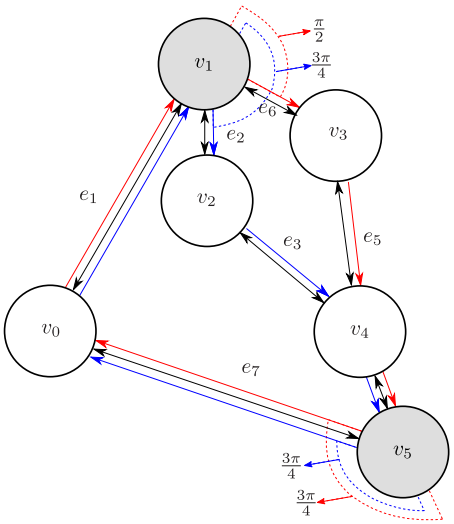


Fig. 5. Example of optimal (in red) and sub-optimal (in blue) solutions. The angles at the service nodes are highlighted by dotted lines.

of variables which are involved in the service graph-based formulation in (15) is significantly reduced compared to the previous one since it focuses only on the nodes to serve, for which it generally holds $|\mathcal{V}_s| \ll |\mathcal{V}|$. The sub-optimality of the solution arises from the fact that, when solving the local problem (10) for each edge (v_s, v_t), the turning costs to *exit* node v_t are not considered, i.e., the turning costs to the subsequent part of the path are not taken into account. This is mitigated by the turning costs $\hat{C}_j^{t_s}$ in (13) which, however, do not ensure that the optimal path in the orientation graph is obtained.

Figure 5 shows the sub-optimal and optimal solutions for the example scenario in Figure 4. The considered sub-optimal and optimal solutions correspond to cycles $\{e_1, e_2, e_3, e_4, e_7\}$ (highlighted with red arrows) and $\{e_1, e_5, e_6, e_4, e_7\}$ (highlighted with blue arrows), respectively. In the figure, the dotted lines, colored in red and blue, represent the turning angles at service nodes for the optimal and sub-optimal solutions, respectively. We can notice that the turning angle at node v_1 is higher $(3\pi/4)$ for the sub-optimal solution compared to the optimal one $(\pi/2)$, while it is the same $(3\pi/4)$ at node v_5 for both solutions. This is due to the fact that the sub-optimal method selects the shortest paths ignoring the turning costs at the service nodes, while the optimal solution, which in general does not consist of solely shortest paths, compromises between minimizing the path cost and the turning angle at the service nodes. This is the case for the optimal solution depicted in the figure, which, from node v_1 to v_5 follows a different, slightly longer, path than the shortest one (chosen by the sub-optimal solution in blue), but compromises with a much smaller turning angle at node v_1, thus minimizing the overall cycle cost. We now formally provide a bound on the optimality gap with respect to the formulation in (9a).

Theorem 1: Consider the multi-Steiner TSP formulation based on the orientation graph in (9a) with optimal cost C^* and the formulation based on the service graph in (15) with optimal cost $\hat{C}^*$, then the relative optimality gap can be bounded as follows

$$\frac{\hat{C}^* - C^*}{C^*} \leq 2\pi \, \frac{\beta \overline{c}^t}{\alpha \underline{c}^e + \gamma \, \underline{c}^s}, \tag{16}$$

with $\overline{c}^t = \max_{r_j \in \mathcal{R}} c_j^t$ the maximum turning unit cost, $\underline{c}^e = \min_{r_j \in \mathcal{R}} c_j^e$ the minimum edge traversing unit cost, and $\underline{c}^s = \min_{r_j \in \mathcal{R}, v_i \in \mathcal{V}_s} c_{i,j}^s$ the minimum service cost.

Proof: A bound on the relative optimality gap can be obtained by maximizing the difference at numerator $\widetilde{C} = \hat{C}^* - C^*$, while minimizing the denominator term C^*. Let us define the following variables related to the solution of (9a)

$$E^* = \alpha \, C_{\max}^{e*} + \alpha \sum_{r_j \in \mathcal{R}} C_j^{e*}$$
$$T_t^* = \beta \, C_{\max}^{t_t*} + \beta \sum_{r_j \in \mathcal{R}} C_j^{t_t*}$$
$$T_s^* = \beta \, C_{\max}^{t_s*} + \beta \sum_{r_j \in \mathcal{R}} C_j^{t_s*}$$
$$S^* = \gamma \, C_{\max}^{s*} + \gamma \sum_{r_j \in \mathcal{R}} C_j^{s*}$$

where $C_{\max}^{(\cdot)*}$ is the respective component in $C_{\max}^*$, and the equivalent variables $\hat{E}^*$, $\hat{T}_t^*$, $\hat{T}_s^*$, $\hat{S}^*$ for the solution of (15). The costs $\hat{C}^*$ and C^* can be rewritten as

$$C^* = E^* + T_t^* + T_s^* + S^*, \quad \hat{C}^* = \hat{E}^* + \hat{T}_t^* + \hat{T}_s^* + \hat{S}^*.$$

Since the service based-formulation in (15) does not globally optimize for the turning costs $\hat{T}_s^*$ at the nodes to serve, we consider its worst case scenario to characterize the bound on $\widetilde{C}$. In particular, the worst case arises when the angle $\hat{\varphi}_{p,q,j}$ between subsequent paths $\mathcal{P}_{p,j}$ and $\mathcal{P}_{q,j}$ (obtained according to (10)) is always maximum, i.e., $\hat{\varphi}_{p,q,j} = \pi \ \forall (\hat{e}_p, \hat{e}_q) \in \hat{\mathcal{E}}_c, r_j \in \mathcal{R}$, and the maximum cycle serves all the nodes $v_i \in \mathcal{V}_s$ with maximum unit turning cost $\overline{c}^t$. It follows

$$\hat{T}_s^* \leq 2\pi\beta \, \overline{c}^t |\mathcal{V}_s|.$$

Moreover, by construction it holds that the orientation-based formulation achieves lower or equal performance than the service-based one, i.e., $C^* \leq \hat{C}^*$, implying that

$$E^* + T_t^* + T_s^* + S^* \leq 2\pi\beta \, \overline{c}^t |\mathcal{V}_s| + \hat{E}^* + \hat{T}_t^* + \hat{S}^*. \tag{17}$$

At this point, we can observe that, when $\hat{\varphi}_{p,q,j} = \pi \ \forall p, q, j$, no optimization is made by the service-based formulation on the turning costs at the service nodes. This implies that the latter formulation will only minimize the remaining terms in the cost function and will achieve their minimum possible values, i.e., by construction the following inequality is verified

$$E^* + T_t^* + S^* \geq \hat{E}^* + \hat{T}_t^* + \hat{S}^*. \tag{18}$$

Considering (18) in (17), it follows $T_s^* \leq 2\pi\beta\overline{c}^t |\mathcal{V}_s|$. The difference $\widetilde{C}$ is thus maximized when $T_s^* = 0$, i.e., no turning costs on the service nodes are present in the orientation graph-based solution, leading to

$$\widetilde{C} \leq 2\pi\beta\overline{c}^t |\mathcal{V}_s|. \tag{19}$$

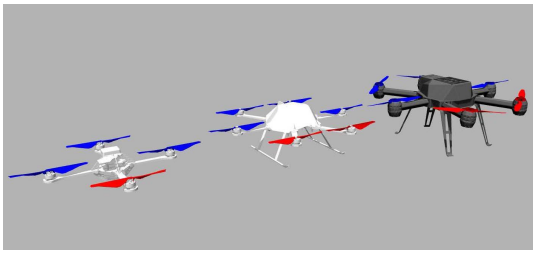


Fig. 6. Aerial vehicles involved in the simulation setup. From the left: Hummingbird, Firefly, Neo11.

As far as the minimization of C^* is concerned, we observe that its lower bound is obtained when there are no turning costs at the service nodes, i.e., $T_s^* = 0$, and the robot with lowest edge traversing and service costs is chosen for each service node, leading to $C^* \geq (\alpha \underline{c}^e + \gamma \underline{c}^s)|\mathcal{V}_s|$. The relative optimality gap is thus bounded as in (16), completing the proof. ■

Remark 2: The above theorem states that the bound on the relative optimality gap depends on the weighted ratio between the maximum unit turning cost and the minimum unit cost for traversing an edge and cost for servicing a node. This implies that the lower the relative cost of turning operations, the more convenient the service graph. Moreover, the bound on the absolute optimality gap in (19) shows that the lower the number of service nodes, $|\mathcal{V}_s|$, the lower the optimality loss using the sub-optimal formulation, proving that the latter is particularly convenient when $|\mathcal{V}| \gg |\mathcal{V}_s|$. Note that, as shown in the numerical results in Section VI, the derived bound is rather conservative and comparable results in terms of optimality are generally achieved by the orientation-based and the service-based formulations.

VI. Simulation Results

In this section, we provide simulation results to prove the effectiveness of the proposed formulations.

A. Setup Description

Inspired by the needs of employing robotics in PA settings, we considered a simulated orchard, shown in Figure 1, in which three aerial robots ($m = 3$) perform targeted inspection tasks. Such inspections can be aimed at monitoring any relevant field parameters, such as health status, soil condition and yield [57]. The following heterogeneous robots, shown in Figure 6, were deployed: 1) AscTec Hummingbird (Figure 6-left), with smallest size, 2) AscTec Firefly (Figure 6-middle), with medium size, 3) Neo11 (Figure 6-right), with largest size. Each aerial robot is equipped with the same camera mounted on the bottom part of the drone by means of a rotational joint.

Each inspection task requires the assigned robot to navigate through the plants to reach the target one and to hover for a certain amount of time in correspondence of that plant to collect relevant agronomic data. In particular, during the hovering, a scan of the plant, from the bottom to its top part, is made through the camera sensor. Such a scan could be aimed, for example, to detect pest infestations on the plants, which generally require inspections at heights between 1.5 and 3 meters [58]. Note that the proposed formulations can be adapted to perform any agricultural task in permanent crops, and the inspection task considered hereby is only a representative example for demonstration purposes.

TABLE III

Temporal (c_j^e, c_j^t, $c_{i,j}^s$ $\forall v_i \in \mathcal{V}_s$) and Energy ($\epsilon_j^e$, ϵ_j^t, ϵ_j^s) Cost Coefficients as Well as Energy Capacity (K_j) of the Three Aerial Robots Hummingbird, Firefly and Neo11

Robot	c_j^e [s/m]	c_j^t [s/rad]	c_j^s [s]	ε_j^e [1/s](%)	ε_j^t [1/s](%)	ε_j^s [1/s](%)	K_j (%)
Humm.	0.48	0.09	30	0.05	0.005	35	100
Firef.	0.59	0.12	30	0.01	0.001	21	100
Neo11	0.89	0.44	30	0.001	0.0001	5	100

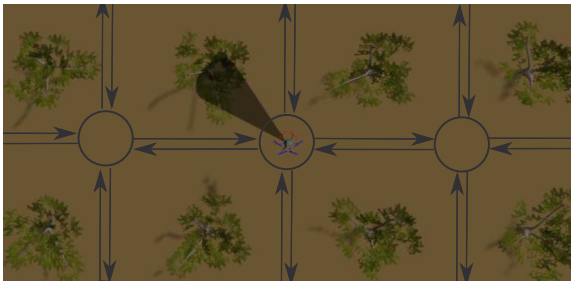


Fig. 7. Representation of an inspection task. The respective portion of the field graph is also shown in gray.

Numerical values for the robots' cost coefficients c_j^e, c_j^t, $c_{i,j}^s$, ϵ_j^e, ϵ_j^t, ϵ_j^s, K_j, $j = 1, 2, 3$ $\forall v_i \in \mathcal{V}_s$, defined in Section IV, are reported in Table III. Costs $c_{i,j}^s$ were set equal for all service nodes $v_i \in \mathcal{V}_s$ and for all robots $r_j \in \mathcal{R}$ as we assume that each plant must be scanned for the same amount of time. Temporal unit costs c_j^e and c_j^t are expressed in [s/m] and [s/rad], respectively, while temporal service costs $c_{i,j}^s$ are expressed in [s]. Energy unit costs ϵ_j^e, ϵ_j^t and ϵ_j^s are all expressed in [1/s] and represent the percentage of consumed energy in the time unit. We assume that every robot is fully charged ($K_j = 100\%$, $j = 1, 2, 3$). The Hummingbird is the smallest and fastest robot among the considered ones, leading to the highest efficiency in terms of temporal unit costs (lowest values for c_j^e and c_j^t), but the lowest energetic efficiency (highest values for ϵ_j^e, ϵ_j^t, ϵ_j^s). In contrast, Neo11 is the biggest and slowest robot, but relies on the most powerful battery which provides it with the greatest autonomy. Intermediate cost coefficients are considered for the Firefly robot. Weights $\alpha = 1$, $\beta = 40$, $\gamma = 1$ were used for the objective functions to prioritize the minimization of the turning costs.

Concerning the field graph, we considered orchards with $N \times N$ planting patterns, resulting in square grid topology, in which the trees are distant 5 meters from one another and with variable altitude. The location in the middle between four trees is associated with a node $v_i \in \mathcal{V}$ and the inspection is made on the top left tree. An example is reported in Figure 7 where the robot's location is associated with a node and it is shown while performing an inspection task. The figure also shows the respective portion of the field graph in gray. Passages between these locations are associated with edges

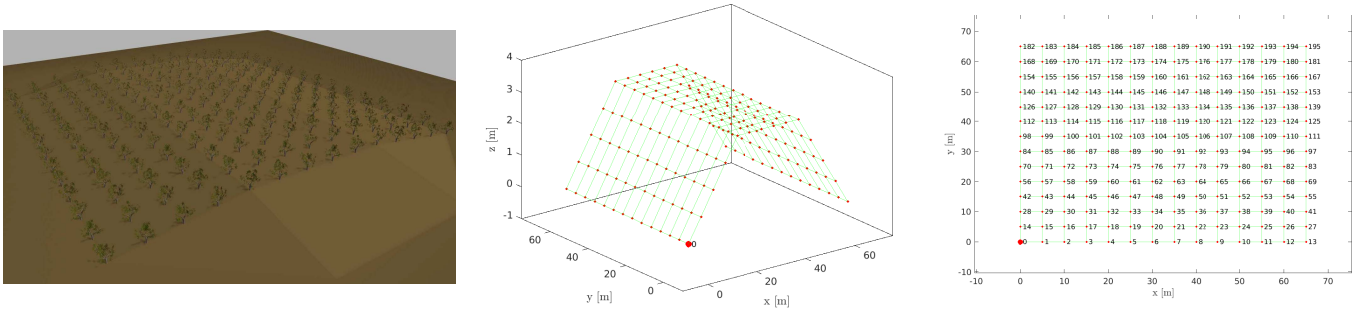


Fig. 8. Representation of the orchard in GAZEBO simulator (left) and the respective field graph in 3D (middle) and in 2D from the top view (right). Only the depot node is highlighted in the middle figure for the sake of readability.

$e_k \in \mathcal{E}$. The depot node is labeled as 0 and is at zero altitude. From there, the field slopes up to reach a plain surface at 3 meters of altitude, corresponding to the maximum altitude of the orchard, and then slopes down again to zero altitude. The altitude h_k of node v_k in row i and column j of the square grid is given by:

$$h_k = \begin{cases} \dfrac{3}{\lfloor N/3 \rfloor}(j-1), & j \leq \lfloor N/3 \rfloor \\ 3, & \lfloor N/3 \rfloor < j \leq N - \lfloor N/3 \rfloor \\ 3 - \dfrac{3}{\lfloor N/3 \rfloor} & \\ \quad \times (j - N + \lfloor N/3 \rfloor), & j > N - \lfloor N/3 \rfloor. \end{cases}$$

Figure 8 shows an example of the simulated hazelnut orchard (left) and the respective field graph with $N = 14$ (middle). The same topology in a two dimensional perspective from above is also shown (right). Without loss of generality, this perspective will be used to show the robots' routes. Concerning the selection of the plants to inspect, different subsets $\mathcal{V}_s$ of $\mathcal{V}$ with varying size have been considered as detailed in the following. We considered the case in which $|\mathcal{V}_s| \ll |\mathcal{V}|$ is verified, which is typical for PA applications.

All software components were developed in MATLAB interfaced with IBM CPLEX 12.10 solver and with ROS middleware. The latter was then interfaced with GAZEBO simulator to reproduce realistic aerial vehicles motion as well as the orchard. In accordance with the vision of PANTHEON project aiming at building an agricultural equivalent of an industrial SCADA system, a central unit was in charge of computing the solutions of the optimization problems. The input for both optimal and sub-optimal formulations consists of: *i)* the field graph $\mathcal{G}$, modeling the topology of the orchard, *ii)* the set of vertices $\mathcal{V}_s$, where inspection tasks are required, and *iii)* the temporal and energy coefficients for each robot. The output of each optimization problem consists of: *i)* the sequence of edges to traverse for each robot, and *ii)* the set of locations to inspect. These sequences are sent to a ROS controller node [59] generating the motion control commands for the simulated aerial vehicles. A workstation with processor Intel Xeon E5-2650L v4 with 35 MB cache and 1.70 GHz was used for computation.

B. Comparative Results

We carried out an extensive simulation campaign composed of several tests involving fields of various dimensions and different numbers of nodes to serve. Moreover, a greedy solution and two optimization-based solutions to solve the multi-Steiner TSP problem were implemented for a comparative analysis.

The greedy algorithm is summarized in Algorithm 2. Briefly, the basic idea is to iteratively assign the service nodes to the robots that are the most efficient in reaching and serving them, while ensuring to travel back to the depot, until all service nodes are assigned or the energy of the robots is exhausted. To build the greedy solution, the minimum cost paths between service nodes, contained in the service graph $\hat{\mathcal{G}}$, are exploited. The first phase of the algorithm consists in initializing the sets $\tilde{\mathcal{V}}_s$ and $\mathcal{Q}_j$, $\forall r_j \in \mathcal{R}$ (lines 2.1-2.2), comprising the unassigned service nodes and the edges in the route of robot r_j, as well as the residual energy $\tilde{K}_j$ (line 2.3) of each robot r_j. A flag variable denoting if any feasible solution is found is also initialized (line 2.4). The second phase of the algorithm focuses on assigning the service nodes to the robots and building their routes $\mathcal{Q}_j, \forall r_j \in \mathcal{R}$. This is achieved in an iterative manner (line 2.1) until all nodes are assigned, i.e., $\tilde{\mathcal{V}}_s = \emptyset$, or no node can be assigned due to a lack of energy. In the latter case, the algorithm terminates and returns unfeasible solution. At each iteration, the feasible solution flag is set to false (line 2.2) and, for all the couples composed of a remaining node to visit $v_h \in \tilde{\mathcal{V}}_s$ and a robot $r_j \in \mathcal{R}$, the energetic costs to reach the node to visit v_h and then to return to the depot v_0 are computed (lines 2.4-2.6). If the robot residual energy is sufficient to cover such costs, the solution feasibility is set (line 2.8) and the cost $C_{h,j}$ for robot j to serve v_h is stored. At this point, the couple composed of robot r_j and node to serve v_h with minimum cost is assigned and the solution is updated accordingly (lines 2.12-2.15). Finally, all the robots are required to travel back to the depot (lines 3.1-3.5).

As far as the optimization-based solutions are concerned, as in [46], these consist in pre-assigning the subset of nodes to visit to each robot. In particular, they are based on the following steps: *i)* partition of the set of vertices to visit $\mathcal{V}_s$ in m sets, which differ for the two baselines, *ii)* assignment of each of the m subsets to a robot and *iii)* solution of a single-Steiner TSP problem for each robot. If all single-Steiner TSP problems are

Algorithm 2 Greedy Algorithm

Require: Service graph $\hat{\mathcal{G}}$, Service nodes $\mathcal{V}_s$, Set of robots $\mathcal{R}$, coefficients $\alpha, \beta, \gamma, c_j^k, c_{i,j}^s$ with $k = e, t$, $\forall v_i \in \mathcal{V}_s$, ϵ_j^k with $k = e, t, s$ $\forall r_j \in \mathcal{R}$

Phase 1 - Initialization

$\tilde{\mathcal{V}}_s \leftarrow \mathcal{V}_s$
$\mathcal{Q}_j \leftarrow \emptyset, \ \forall r_j \in \mathcal{R}$
$\tilde{K}_j \leftarrow K_j, \ \forall r_j \in \mathcal{R}$
feasible solution $\leftarrow true$

Phase 2 - Compute routes $\mathcal{Q}_j$, $\forall R_j \in \mathcal{R}$

while $\tilde{\mathcal{V}}_s \neq \emptyset \wedge$ feasible solution **do**
  feasible solution $\leftarrow false$
  for each $v_h \in \tilde{\mathcal{V}}_s, \ r_j \in \mathcal{R}$ **do**
    $v_l \leftarrow$ get last node$(\mathcal{Q}_j)$
    $E_{h,j} \leftarrow$ energy between nodes(r_j, v_l, v_h)
    $E_{0,j} \leftarrow$ energy between nodes(r_j, v_h, v_0)
    if $E_{h,j} + E_{0,j} \leq \tilde{K}_j$ **do**
      feasible solution $\leftarrow true$
      $C_{h,j} \leftarrow$ cost between nodes(r_j, v_h, v_0)
    end if
  end for
  $v_{h^*}, r_{j^*} \leftarrow$ argmin$(C_{h,j})$
  if feasible solution **then**
    $(\mathcal{Q}_{j^*}, \tilde{K}_{j^*}) \leftarrow$ update robot$(r_{j^*}, v_{h^*}, E_{h^*,j^*})$
    $\tilde{\mathcal{V}}_s \leftarrow \tilde{\mathcal{V}}_s \setminus \{v_h\}$
  end if
end while

Phase 3 - Return to depot

if $\tilde{\mathcal{V}}_s \equiv \emptyset$ **then**
  for each $r_j \in \mathcal{R}$ **do**
    $\mathcal{Q}_j \leftarrow$ add edge to node(v_0)
  end for
end if

feasible, a feasible solution for the multi-Steiner TSP problem is built. Two partitioning algorithms were considered for the two methods: *i)* k-means [60] and *ii)* random.

From here on, the described baseline approaches with k-means and random partitioning algorithms will be referred to as k-means and random methods, respectively, while we will refer to the optimization problems in (9a) and in (15) as optimal and sub-optimal methods, respectively. We selected these methods for comparison as k-means naturally tries to group together points of interest that are spatially close to each other, while random is a basic baseline which is necessarily required to be overcome by the proposed approaches.

Plots in Figure 9 summarize the results of the simulation campaign. We considered field sizes ranging from 10×10 to 14×14, and for each size we considered an increasing percentage of trees to be inspected, ranging from 2% to 5%, of the total number. In case the percentage does not lead to an integer value, we round up to the closest integer. For all the methods, each simulation case was executed 10 times, considering a different random selection of trees to be inspected at each execution. For example, in the 14×14 field, an inspection of 3% of trees means that 6 trees ($|\mathcal{V}_s| = 6$) are randomly selected from the field 10 times, defining 10 different scenarios; the same 10 scenarios were used for each of the five methods. In all the plots the results related to optimal, sub-optimal, greedy, k-means and random methods are represented in light blue, green, red, yellow and purple, respectively. Average values are given by the heights of the bars, while black lines represent standard deviations. In the figure, the top left plot shows the average temporal costs C^*, $\hat{C}^*$, C_g^*, C_k^* and C_r^* along with the standard deviation achieved by the optimal, sub-optimal, greedy, k-means and random methods, respectively. The top right plot reports the average relative optimality gaps of the temporal costs evaluated for the non-optimal methods (i.e., sub-optimal, greedy, k-means and random) with respect to the optimal solution. We denote with $(\hat{C}^* - C^*)/C^*$, $(C_g^* - C^*)/C^*$, $(C_k^* - C^*)/C^*$, $(C_r^* - C^*)/C^*$, the relative optimality gap for the sub-optimal, greedy, k-means and random methods, respectively. The bottom left plot depicts the number of times the proposed methods failed to find a solution for the simulation cases. Only k-means and random are reported since the others always succeed in finding feasible solutions. The bottom right plot shows the average computational times Δ, $\hat{\Delta}$, Δ_g, Δ_k and Δ_r along with the standard deviations for the optimal, sub-optimal, greedy, k-means and random methods, respectively. In particular, the top left plot shows that, as expected, the average temporal cost for all methods increases with the field size and percentage of plants to inspect. Moreover, the plot highlights that similar average temporal costs are achieved by the sub-optimal method compared to the optimal one, while higher costs are obtained when using the remaining sub-optimal approaches. This trend is particularly evident in the top right plot where the average relative optimality gap is shown to be stable across different field sizes and nodes to inspect with the sub-optimal method (in green) and always lower than 4.83%. In particular, the sub-optimal method achieved the lowest optimality gap compared to the other methods. These results confirm the effectiveness of the proposed sub-optimal formulation against the baseline solutions. Moreover, by comparing the actual relative gap of the sub-optimal solution with the relative upperbound (362.81%), it can be also noticed that the bound is largely satisfied, confirming its conservativeness. This is motivated by the fact that the upperbound is obtained considering an unrealistic scenario in which the optimal solution *i)* only deploys the robot with lowest edge traversing and service costs to inspect all nodes in $\mathcal{V}_s$, *ii)* does not form a cycle, *iii)* involves zero turning costs at the service nodes and *iv)* has *each* path from an inspected node to the subsequent one composed of a *single*, minimum cost, edge. As far as the greedy method is concerned, relative optimality gap lower than 30.06% is obtained, but the numerical analysis indicates that this gap increases as the field size and nodes to inspect increase. This is motivated by the fact that the higher the number of service nodes to be assigned to the robots, the higher the chances that the greedy approach builds an inefficient solution. High optimality gaps are obtained instead in all cases by k-means and random methods. Note that the low optimality performance of these methods was expected, as they are not

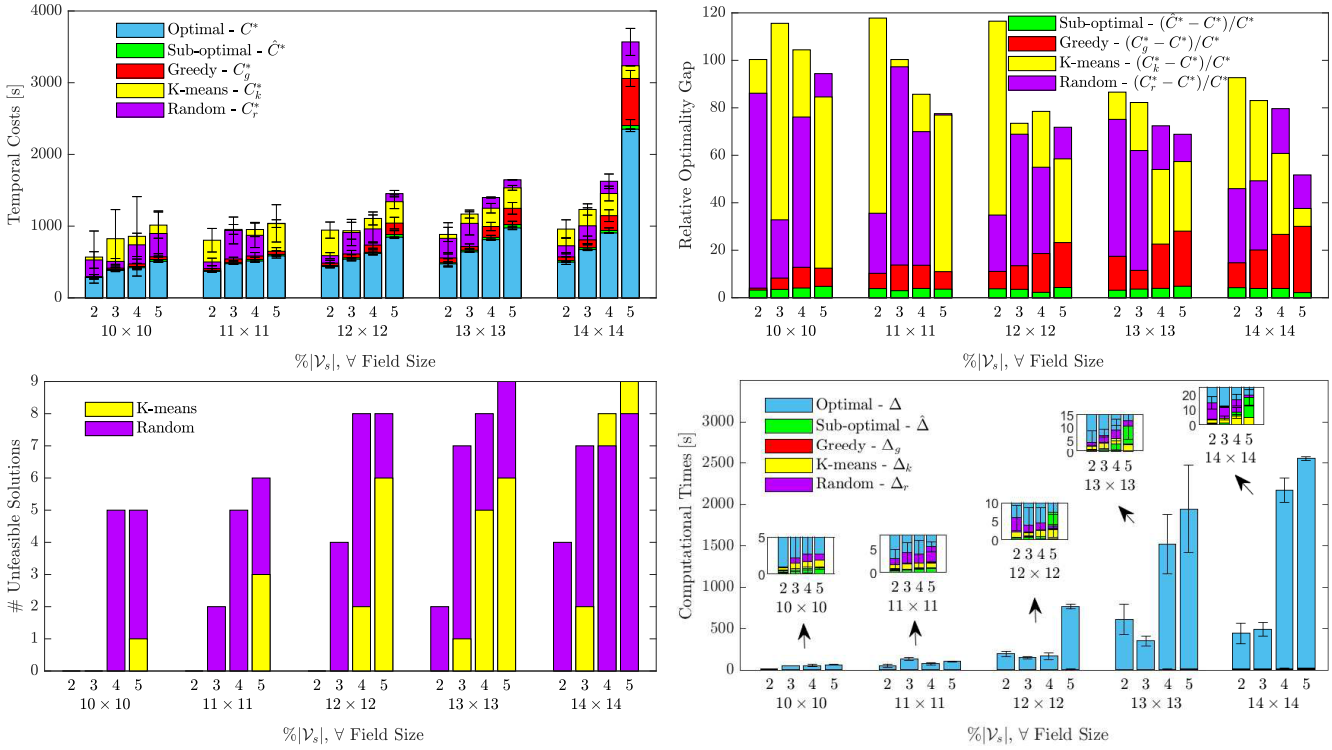


Fig. 9. Simulation results for the optimal (light blue), sub-optimal (green), greedy (red), k-means (yellow) and random (purple) methods considering fields sizes from 10×10 to 14×14 and increasing percentage of trees to be inspected ($\%|Vs|$), from 2% to 5% of the total number. Top left plot shows average and standard deviation of temporal costs C^*, $\hat{C}^*$, C_g^*, C_k^* and C_r^*. Top right plot shows the average relative optimality gaps of the temporal costs. Bottom left plot shows the number of unfeasible solutions. Bottom right plot shows average and standard deviation of computational times Δ, $\hat{\Delta}$, Δ_g, Δ_k and Δ_r.

specialized for the problem at hand and, therefore, fail to solve it efficiently. In fact, the node pre-allocation performed by the partitioning algorithms does not take into account either the objective function in (9a) or the energy constraints of the robots. However, we would had to resort to these comparison methods since, as discussed in the Related work section, no other method exists at the state of the art that can efficiently address our problem. The observations regarding the optimality performance of the different methods are also confirmed by the number of unfeasible solutions in the bottom right plot. More specifically, the sub-optimal and greedy methods managed to find a solution for each execution, while, for the random and k-means methods, the higher the field size and nodes to serve, the higher the number of failures recorded, achieving $9/10$ unfeasible solutions with field sizes 13×13 and 14×14 and 5% of nodes to visit, respectively. Regarding the computational aspects, the bottom left plot demonstrates the significant difference in computational time of the optimal solution compared to the non-optimal ones. In particular, the computational time Δ for the optimal method increases rapidly as the field size and the number of nodes to serve increase, passing from the order of seconds for the 10×10 field case to tens of minutes for the 14×14 field. Zoomed views for each field size are provided to clearly show the computational time for the non-optimal methods. Computational time $\hat{\Delta}$ in the order of seconds is always recorded for the sub-optimal method, achieving maximum average value equal to 18.01 s for 14×14 field with 5% of nodes to serve. Similar computational times are generally recorded by k-means and random baselines, achieving on average computational times equal to 2.90 s and to 6.93 s, respectively, against 3.05 s for the sub-optimal method. Finally, much lower computational times equal to 0.09 s on average are obtained for the greedy method, which, as opposed to the others, does not need to solve any NP-hard optimization problem. To summarize, the simulation campaign allows to numerically corroborate the effectiveness of the proposed formulations against three baseline solutions and show the trade-off between optimality and computational time of the sub-optimal formulation.

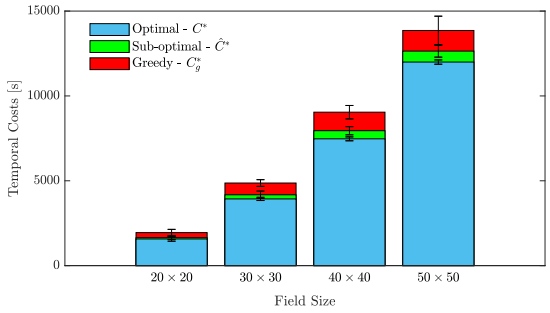
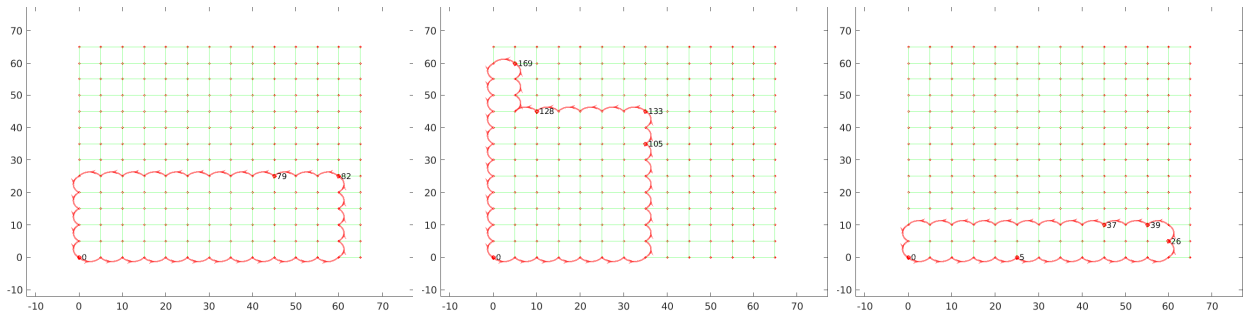
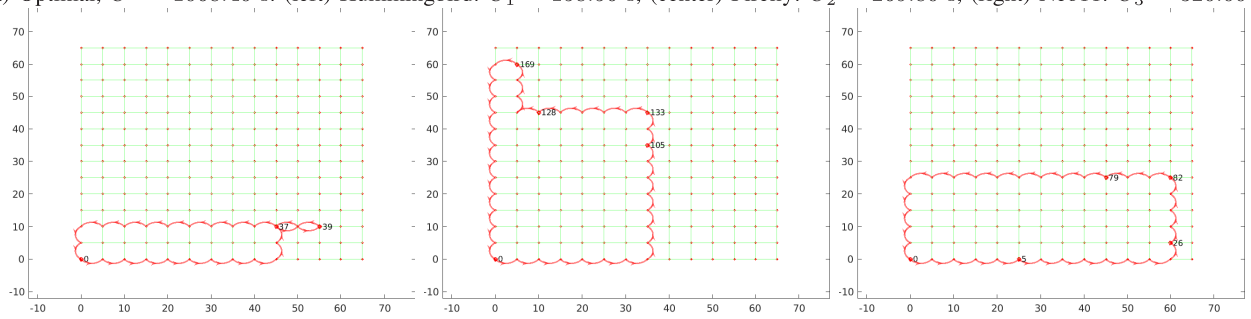


Fig. 10. Temporal costs C^*, $\hat{C}^*$, C_g^* for the optimal (light blue), sub-optimal (green), greedy (red) methods considering fields sizes from 20×20 to 50×50 with 3% of the total number of trees to be inspected.

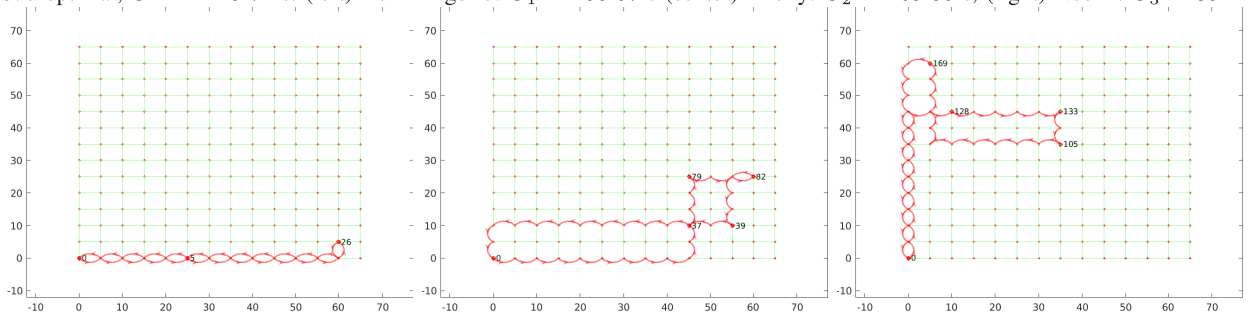
Based on the same methods as in the above, we performed a scalability analysis with field sizes ranging from 20×20 to 50×50 and requiring 3% of the total number of trees to be inspected. Figure 10 shows the results obtained in this analysis.



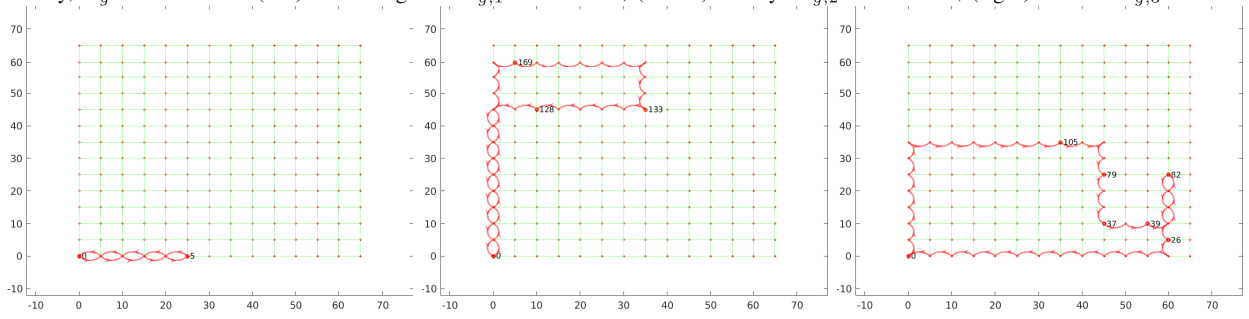
(a) Optimal, $C^* = 1068.40$ s: (left) Hummingbird: $C_1^* = 158.56$ s, (center) Firefly: $C_2^* = 269.80$ s, (right) Neo11: $C_3^* = 320.00$ s.



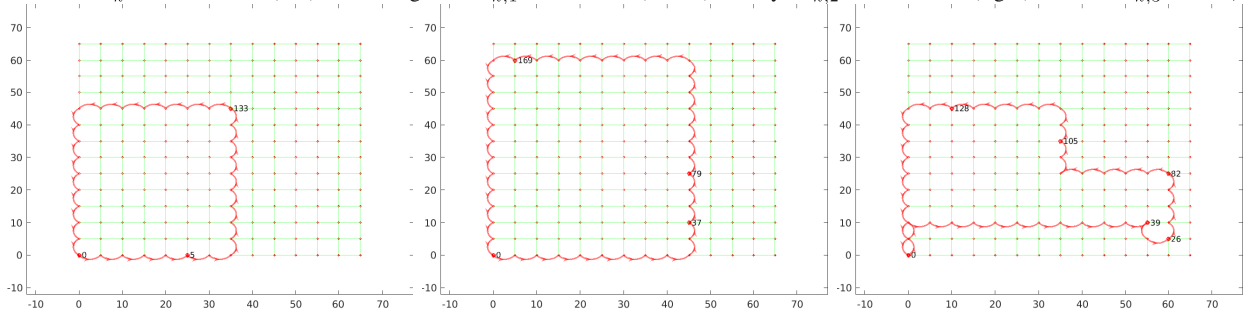
(b) Sub-optimal, $\hat{C}^* = 1129.71$ s: (left) Hummingbird: $\hat{C}_1^* = 150.67$ s (center) Firefly: $\hat{C}_2^* = 269.80$ s, (right) Neo11: $\hat{C}_3^* = 354.24$ s.



(c) Greedy, $C_g^* = 1363.87$ s: (left) Hummingbird: $C_{g,1}^* = 145.02$ s, (center) Firefly: $C_{g,2}^* = 273.08$ s, (right) Neo11: $C_{g,3}^* = 472.78$ s.



(d) K-means, $C_k^* = 1401.29$ s: (left) Hummingbird: $C_{k,1}^* = 65.31$ s, (center) Firefly, $C_{k,2}^* = 232.26$ s, (right) Neo11, $C_{k,3}^* = 551,73$ s.



(e) Random, $C_r^* = 1484.12$: (left) Hummingbird: $C_{r,1}^* = 153.76$ s, (center) Firefly: $C_{r,2}^* = 236.52$ s, (right) Neo11: $C_{r,3}^* = 546.86$ s.

Fig. 11. Examples of solutions obtained by the optimal (a), sub-optimal (b), greedy (c), k-means (d), and random (e) methods.

The average (heights of the bars) temporal costs C^*, $\hat{C}^*$, C_g^* (obtained over 5 instances) along with the standard deviation (black lines) achieved by the optimal (light blue), sub-optimal (green), and greedy (red) methods, respectively, are shown. No results concerning k-means and random methods are provided since they did not manage to find solutions in the considered cases, as coherent with the results obtained for field size equal to 13×13 and 14×14. On the contrary, the optimal, sub-optimal and greedy methods always found a solution. The obtained results confirm the behavior observed with smaller field graphs, i.e., *i)* the average temporal cost for all methods increases with field size and *ii)* the sub-optimal method always achieves a better cost, i.e., closer to the optimal one, than the greedy solution. To cope with the NP-hardness of the optimal and sub-optimal methods, in these simulations we relaxed the search of the solver for the exact optimum and allowed 10% optimality margin.

C. Illustrative Example

We now illustrate the behaviour of the five implemented methods to an example scenario. We considered a 14×14 field with 5% of nodes to be inspected, implying $|\mathcal{V}_s| = 10$. For this example, $\mathcal{V}_s = \{5, 26, 37, 39, 79, 82, 105, 128, 133, 169\}$. The accompanying video shows the execution of the optimal plans for the three aerial robots according to architecture described in Section VI-A. Figure 11 reports the solutions obtained by the optimal (Figure 11-a, first row), sub-optimal (Figure 11-b, second row), greedy (Figure 11-c, third row), k-means (Figure 11-d, fourth row) and random (Figure 11-e, fifth row) methods. Each column is associated with a robot, i.e., from the left there are Hummingbird, Firefly and Neo11. The route planned for each robot (red edges) as well as the set of nodes to serve (red labeled dots) are reported. In the figures, only the depot and the inspected trees are labeled for the sake of readability. The optimal solution achieves overall cost $C^* = 1068.40$ s and requires Hummingbird to inspect nodes 79 and 82 in 158.56 s using 78.52% of its energy capacity; Firefly to inspect nodes 105, 128, 133 and 169 in 269.80 s using 85.91% of its total energy capacity; Neo11 to inspect 5, 26, 37 and 39 in 320.00 s using 20.14% of its total energy capacity. The figure makes evident that robots are deployed to inspect groups of trees which are relatively close to one another. Moreover, the optimal solution maximizes the number of nodes that the fastest robots (Hummingbird and Firefly) inspect, given their total energy capacity K_j and energy consumption. In this regard, Hummingbird and Firefly can inspect at most two and four nodes, respectively, since they require 35% and 21% of the total energy to inspect a single node, respectively. The rest of the nodes to inspect are assigned to the slowest (but with the longest autonomy) robot, Neo11, which requires only 5% of its total energy to inspect a single node. As far as the sub-optimal method is concerned, cost $\hat{C}^* = 1129.71$ s is obtained. In particular, it assigns Hummingbird to inspect nodes 37 and 39 in 150.67 s using 78.52% of its energy; Firefly to inspect nodes 105, 128, 133 and 169 in 269.80 s using 85.91% of its total energy; Neo11 to inspect 5, 26, 79 and 82 in 354.24 s using 20.14%. The sub-optimal solution maximises the number of nodes assigned to the fastest robots (i.e., Hummingbird and Firefly), but unlike the optimal solution, some robots may follow sub-optimal routes. The non optimalility of this method is a consequence of the fact that this method disposes of pre-assigned shortest paths that do not necessarily belong to the an optimal solution of the problem.

Regarding the greedy algorithm, it achieves cost $C_g^* = 1363.87$ s. In particular, it assigns Hummingbird to inspect nodes 5 and 26 in 145.02 s using 77.76% of its energy; Firefly to inspect nodes 37, 39, 79 and 82 in 273.08 s using 86.14% of its total energy; Neo11 to inspect 105, 128, 133 and 169 in 472.78 s using 20.25%. The greedy solution, alike the sub-optimal one, maximizes the number of nodes assigned to the fastest robots, i.e., Hummingbird and Firefly (first and second columns), within their energetic limitations and assigns sub-optimal routes to the Firefly and Neo11 (second and third columns), showing intersecting routes.

For the K-means solution, cost $C_k^* = 1401.29$ s is obtained. In this case, Hummingbird inspects node 5 in 65.31 s using 37.51% of its energy; Firefly inspects nodes 128, 133 and 169 in 232.26 s using 64.90% of its total energy; Neo11 inspects nodes 26, 37, 39, 79, 82 and 105 in $551,73$ s using 30.22% of its total energy. K-means method deploys robots to inspect nodes which are relatively close to one another. This feature alone, although in common with the optimal solution, does not generally guarantee the optimality of the solution, as it can be seen comparing the k-means solution cost with the optimal one. This is due to the fact that the partitioning algorithm does not take into account the objective function in (9a) (as shown by the results in Figure 9). Furthermore, the number of nodes assigned to each robot is randomly determined, meaning that the fastest robots could be energetically underutilized or overutilized. In the figure, both Hummingbird and Firefly are underused because they inspect a single node and three nodes, respectively, instead of two and four nodes as done in the optimal case, leading to overusing the slowest robot, Neo11.

Finally, with regard to the random method, it obtains cost $C_r^* = 1484.12$ s and requires Hummingbird to inspect nodes 5 and 133 in 153.76 s using 78.02% of its energy; Firefly to inspect nodes 37, 79 and 169 in 236.52 s using 65.10% of its total energy; Neo11 to inspect 26, 39, 82, 105 and 128, in 546.86 s using 25.23% of its total energy. The random pre-assignment strategy generally leads robots to inspect groups of nodes that are scattered on the field, as it can be observed from the figure. Similarly to the k-means method, the Firefly is energetically underused, inspecting three nodes instead of four and resulting in an overuse of the slowest robot, Neo11. We reiterate that k-means and random methods were only selected as baselines for comparison, since no other method is available at the state of the art to deal with the problem at hand. For this reason, low performance in terms of optimality was expected.

VII. Conclusion

In this work, we proposed a novel formulation to plan optimal routes for multiple mobile robots to perform agricultural tasks in large-scale PA settings. Differently from

existing works, we introduced a field model able to account for maneuvering costs with general topologies. Building on this, we formalized a multi-Steiner TSP problem in which turning costs on arbitrary field topologies as well as energy capacity constraints are taken into account. The total time to execute all the tasks, as well as the cumulative execution times of the robots have been minimized. Moreover, we proposed a sub-optimal formulation to mitigate the computational load by relaxing the optimization of the maneuvering costs at the locations where agricultural tasks are carried out and provided a formal analysis on the optimality gap. Simulation results with three aerial robots performing inspection tasks in a orchard corroborated the proposed formulations. Remarkably, the proposed formulations can be easily adapted to any operational setting where multiple mobile robots need to perform tasks in assigned locations of the environment. As future work, we plan to overcome the inherent computational issues of the proposed formulations by designing efficient algorithms for solving the proposed problems. Moreover, we plan to validate the formulations on real-world robotic platforms.

REFERENCES

[1] H. M. Jawad, R. Nordin, S. K. Gharghan, A. M. Jawad, and M. Ismail, "Energy-efficient wireless sensor networks for precision agriculture: A review," *Sensors*, vol. 17, no. 8, p. 1781, Aug. 2017.

[2] M. Lippi, N. Bonucci, R. F. Carpio, M. Contarini, S. Speranza, and A. Gasparri, "A YOLO-based pest detection system for precision agriculture," in *Proc. 29th Medit. Conf. Control Autom. (MED)*, Jun. 2021, pp. 342–347.

[3] C. Potena *et al.*, "Suckers emission detection and volume estimation for the precision farming of hazelnut orchards," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Aug. 2020, pp. 285–290.

[4] B. Shivaprasad, M. Ravishankara, and B. Shoba, "Design and implementation of seeding and fertilizing agriculture robot," *Int. J. Appl. Innov. Eng. Manag.*, vol. 3, no. 6, pp. 251–255, 2014.

[5] J. Bengochea-Guevara, J. Conesa-Muñoz, D. Andújar, and A. Ribeiro, "Merge fuzzy visual servoing and GPS-based planning to obtain a proper navigation behavior for a small crop-inspection robot," *Sensors*, vol. 16, no. 3, p. 276, Feb. 2016.

[6] B. H. Y. Alsalam, K. Morton, D. Campbell, and F. Gonzalez, "Autonomous UAV with vision based on-board decision making for remote sensing and precision agriculture," in *Proc. IEEE Aerosp. Conf.*, Mar. 2017, pp. 1–12.

[7] T. Botterill *et al.*, "A robot system for pruning grape vines," *J. Field Robot.*, vol. 34, no. 6, pp. 1100–1122, Sep. 2017.

[8] Y. Edan, D. Rogozin, T. Flash, and G. E. Miles, "Robotic melon harvesting," *IEEE Trans. Robot. Autom.*, vol. 16, no. 6, pp. 831–835, Dec. 2000.

[9] R. F. Carpio *et al.*, "A navigation architecture for Ackermann vehicles in precision farming," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1103–1110, Apr. 2020.

[10] F. A. A. Cheein and R. Carelli, "Agricultural robotics: Unmanned robotic service units in agricultural tasks," *IEEE Ind. Electron. Mag.*, vol. 7, no. 3, pp. 48–58, Sep. 2013.

[11] B. Talebpour, M. B. Eminoğlu, U. Yegül, and U. Türker, "Spatial and temporal variation in an apple orchard," *HortScience*, vol. 54, no. 12, pp. 2182–2187, Dec. 2019.

[12] D. Modina, D. Bianchi, B. Ortuani, A. Mayer, R. Spadaccini, and L. Brancadoro, "Variable rate irrigation in a vineyard and an orchard," in *Proc. Int. Symp. Precis. Manage. Orchards Vineyards*, 2019, pp. 109–116.

[13] R. F. Carpio, J. Maiolini, C. Potena, E. Garone, G. Ulivi, and A. Gasparn, "MP-STSP: A multi-platform Steiner traveling salesman problem formulation for precision agriculture in orchards," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 2345–2351.

[14] K. Srivastava, P. C. Pandey, and J. K. Sharma, "An approach for route optimization in applications of precision agriculture using UAVs," *Drones*, vol. 4, no. 3, p. 58, Sep. 2020.

[15] P. O. Dusadeerungsikul and S. Y. Nof, "A collaborative control protocol for agricultural robot routing with online adaptation," *Comput. Ind. Eng.*, vol. 135, pp. 456–466, Sep. 2019.

[16] T. H. Pham, D. Ichalal, and S. Mammar, "Complete coverage path planning for pests-ridden in precision agriculture using UAV," in *Proc. IEEE Int. Conf. Netw., Sens. Control (ICNSC)*, Oct. 2020, pp. 1–6.

[17] U. Zangina, S. Buyamin, M. N. Aman, M. S. Z. Abidin, and M. S. A. Mahmud, "A greedy approach to improve pesticide application for precision agriculture using model predictive control," *Comput. Electron. Agricult.*, vol. 182, Mar. 2021, Art. no. 105984.

[18] J. Xie, L. R. G. Carrillo, and L. Jin, "Path planning for UAV to cover multiple separated convex polygonal regions," *IEEE Access*, vol. 8, pp. 51770–51785, 2020.

[19] G. E. Just, M. E. Pellenz, L. A. D. P. Lima, B. S. Chang, R. D. Souza, and S. Montejo-Sánchez, "UAV path optimization for precision agriculture wireless sensor networks," *Sensors*, vol. 20, no. 21, p. 6098, Oct. 2020.

[20] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 1, pp. 287–294, Jan. 2013.

[21] K. Yu, A. K. Budhiraja, and P. Tokekar, "Algorithms for routing of unmanned aerial vehicles with mobile recharging stations," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 2018, pp. 5720–5725.

[22] K. Yu, A. K. Budhiraja, S. Buebel, and P. Tokekar, "Algorithms and experiments on routing of unmanned aerial vehicles with mobile recharging stations," *J. Field Robot.*, vol. 36, no. 3, pp. 602–616, 2019.

[23] P. Maini, K. Sundar, M. Singh, S. Rathinam, and P. B. Sujit, "Cooperative aerial–ground vehicle route planning with fuel constraints for coverage applications," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 55, no. 6, pp. 3016–3028, Dec. 2019.

[24] K. Yu, J. M. O'Kane, and P. Tokekar, "Coverage of an environment using energy-constrained unmanned aerial vehicles," in *Proc. IEEE Int. Conf. Robot. Autom.*, Jun. 2019, pp. 3259–3265.

[25] S. G. Manyam, D. W. Casbeer, and K. Sundar, "Path planning for cooperative routing of air-ground vehicles," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 4630–4635.

[26] S. Alamdari, E. Fata, and S. L. Smith, "Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 138–154, Jan. 2014.

[27] L. F. Tiotsop, A. Servetti, and E. Masala, "An integer linear programming model for efficient scheduling of UGV tasks in precision agriculture under human supervision," *Comput. Oper. Res.*, vol. 114, Feb. 2020, Art. no. 104826.

[28] M. Burger, M. Huiskamp, and T. Keviczky, "Complete field coverage as a multi-vehicle routing problem," *IFAC Proc. Volumes*, vol. 46, no. 18, pp. 97–102, Aug. 2013.

[29] K. Sundar and S. Rathinam, "Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems," *J. Intell. Robotic Syst.*, vol. 88, nos. 2–4, pp. 513–526, Dec. 2017.

[30] N. Mathew, S. L. Smith, and S. L. Waslander, "A graph-based approach to multi-robot rendezvous for recharging in persistent tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, Jul. 2013, pp. 3497–3502.

[31] N. Mathew, S. L. Smith, and S. L. Waslander, "Multirobot rendezvous planning for recharging in persistent tasks," *Trans. Robot.*, vol. 31, no. 1, pp. 128–142, 2015.

[32] S. G. Manyam, S. Rasmussen, D. W. Casbeer, K. Kalyanam, and S. Manickam, "Multi-UAV routing for persistent intelligence surveillance & reconnaissance missions," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2017, pp. 573–580.

[33] D. D. Bochtis and C. G. Sørensen, "The vehicle routing problem in field logistics: Part I," *Biosystems Eng.*, vol. 104, no. 4, pp. 447–457, Dec. 2009.

[34] D. D. Bochtis and C. G. Sørensen, "The vehicle routing problem in field logistics: Part II," *Biosystems Eng.*, vol. 105, no. 2, pp. 180–188, Feb. 2010.

[35] A. Khajepour, M. Sheikhmohammady, and E. Nikbakhsh, "Field path planning using capacitated arc routing problem," *Comput. Electron. Agricult.*, vol. 173, Jun. 2020, Art. no. 105401.

[36] M. Spekken and S. de Bruin, "Optimized routing on agricultural fields by minimizing maneuvering and servicing time," *Precis. Agricult.*, vol. 14, no. 2, pp. 224–244, Apr. 2013.

[37] D. Bochtis, H. W. Griepentrog, S. Vougioukas, P. Busato, R. Berruto, and K. Zhou, "Route planning for orchard operations," *Comput. Electron. Agricult.*, vol. 113, pp. 51–60, Apr. 2015.

[38] I. A. Hameed, D. Bochtis, and C. A. Sørensen, "An optimized field coverage planning approach for navigation of agricultural robots in fields involving obstacle areas," *Int. J. Adv. Robotic Syst.*, vol. 10, no. 5, p. 231, May 2013.
[39] H. Seyyedhasani and J. S. Dvorak, "Reducing field work time using fleet routing optimization," *Biosystems Eng.*, vol. 169, pp. 1–10, May 2018.
[40] A. Utamima, T. Reiners, and A. H. Ansaripoor, "Optimisation of agricultural routing planning in field logistics with evolutionary hybrid neighbourhood search," *Biosystems Eng.*, vol. 184, pp. 166–180, Aug. 2019.
[41] O. Ali, B. Verlinden, and D. Van Oudheusden, "Infield logistics planning for crop-harvesting operations," *Eng. Optim.*, vol. 41, no. 2, pp. 183–197, Feb. 2009.
[42] P. Maini, P. Tokekar, and P. B. Sujit, "Visibility-based persistent monitoring of piecewise linear features on a terrain using multiple aerial and ground robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 4, pp. 1692–1704, Oct. 2021.
[43] P. Tokekar, J. V. Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for precision agriculture," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1498–1511, Dec. 2016.
[44] T. C. Thayer, S. Vougioukas, K. Goldberg, and S. Carpin, "Multirobot routing algorithms for robots operating in vineyards," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1184–1194, Jul. 2020.
[45] K. Sundar and S. Rathinam, "Generalized multiple depot traveling salesmen problem—Polyhedral study and exact algorithm," *Comput. Oper. Res.*, vol. 70, pp. 39–55, Jun. 2016.
[46] V. Gabrel, A. R. Mahjoub, R. Taktak, and E. Uchoa, "The multiple Steiner TSP with order constraints: Complexity and optimization algorithms," *Soft Comput.*, vol. 24, no. 23, pp. 17957–17968, Dec. 2020.
[47] R. Matai, S. Singh, and M. L. Mittal, "Traveling salesman problem: An overview of applications, formulations, and solution approaches," in *Traveling Salesman Problem*, D. Davendra, Ed. Rijeka, Croatia: IntechOpen, 2010, ch. 1.
[48] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, Jun. 2006.
[49] D. Rojas Viloria, E. L. Solano-Charris, A. Muñoz-Villamizar, and J. R. Montoya-Torres, "Unmanned aerial vehicles/drones in vehicle routing problems: A literature review," *Int. Trans. Oper. Res.*, vol. 28, no. 4, pp. 1626–1657, Jul. 2021.
[50] S. L. Smith, M. Schwager, and D. Rus, "Persistent monitoring of changing environments using a robot with limited range sensing," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 5448–5455.
[51] K. Deb, "Multi-objective optimization," in *Search Methodologies*. Cham, Switzerland: Springer, 2014, pp. 403–449.
[52] C. Das, A. Becker, and T. Bretl, "Probably approximately correct coverage for robots with uncertainty," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 1160–1166.
[53] A. N. Letchford, S. D. Nasiri, and D. O. Theis, "Compact formulations of the Steiner traveling salesman problem and related problems," *Eur. J. Oper. Res.*, vol. 228, no. 1, pp. 83–92, Jul. 2013.
[54] Y. Xia, M. Zhu, Q. Gu, L. Zhang, and X. Li, "Toward solving the Steiner travelling salesman problem on urban road maps using the branch decomposition of graphs," *Inf. Sci.*, vol. 374, pp. 164–178, Dec. 2016.
[55] K. Magzhan and H. Jani, "A review and evaluations of shortest path algorithms," *Int. J. Sci. Technol. Res.*, vol. 2, no. 6, pp. 99–104, 2013.
[56] G. Laporte, "Fifty years of vehicle routing," *Transp. Sci.*, vol. 43, no. 4, pp. 408–416, Nov. 2009.
[57] K. Srivastava, A. J. Bhutoria, J. K. Sharma, A. Sinha, and P. C. Pandey, "UAVs technology for the development of GUI based application for precision agriculture and environmental research," *Remote Sens. Appl., Soc. Environ.*, vol. 16, Nov. 2019, Art. no. 100258.
[58] M. Lippi, R. F. Carpio, M. Contarini, S. Speranza, and A. Gasparri, "A data-driven monitoring system for the early detection of pest infestations in the precision agriculture of hazelnut orchards," *IFAC-PapersOnLine*, 2022. [Online]. Available: https://www.events.tum.de/custom/media/Agricontrol_22/Agricontrol2022_Program_v11.pdf and https://www.sciencedirect.com/journal/ifac-papersonline/vol/52/issue/30
[59] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "RotorS—A modular Gazebo MAV simulator framework," in *Robot Operating System (ROS)*. Cham, Switzerland: Springer, 2016, pp. 595–625.
[60] L. Wu, M. A. Garcia, D. Puig, and A. Sole, "Voronoi-based space partitioning for coordinated multi-robot exploration," *J. Phys. Agents*, vol. 1, no. 1, pp. 37–44, 2007.

Antonio Furchì received the M.Sc. degree in computer science and automation engineering from Roma Tre University, Rome, Italy, in 2020, where he is currently pursuing the Ph.D. degree in computer science and automation, under the supervision of Prof. Andrea Gasparri. His main research interests include distributed systems, distributed optimization, and precision agriculture.

Martina Lippi (Associate Member, IEEE) received the M.Sc. *(cum laude)* and Ph.D. degrees in information engineering from the University of Salerno, Fisciano, Italy, in 2017 and 2020, respectively. In 2019, she was a Visiting Scholar at the KTH Royal Institute of Technology, Sweden. From November 2020 to June 2022, she was a Post-Doctoral Researcher at Roma Tre University, Italy, where she has been an Assistant Professor, since June 2022. Her research interests include human–robot interaction, multimanipulator systems, and distributed control.

Renzo Fabrizio Carpio received the master's degree in computer science and automation engineering and the Ph.D. degree in automation and computer science from Roma Tre University, Rome, Italy, in 2017 and 2021, respectively. In 2018, he was a Research Fellow at Roma Tre University, where he has been a Post-Doctoral Researcher, since November 2021. His research interests include multirobot control systems, and motion and task planning for multirobot systems in outdoor and unstructured environments.

Andrea Gasparri (Senior Member, IEEE) received the Laurea degree *(cum laude)* in computer science and the Ph.D. degree in computer science and automation from Roma Tre University, Rome, Italy, in 2004 and 2008, respectively. He is currently a Professor with the Department of Engineering, Roma Tre University. He was a recipient of the Italian Grant FIRB Futuro in Ricerca 2008 for the project Networked Collaborative Team of Autonomous Robots funded by the Italian Ministry of Research and Education. He was the Coordinator of the project "PANTHEON" supported by the European Community within the H2020 framework (under grant agreement number 774571) and is currently the Coordinator of the project "CANOPIES" within the H2020 framework (under grant agreement number 101016906). His research interests include robotics, sensor networks, networked multiagent systems, and precision agriculture. From 2017 to 2021, he was an Associate Editor for the IEEE TRANSACTIONS ON CYBERNETICS. Since 2021, he has been an Associate Editor of the IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS.

Open Access funding provided by 'Università degli Studi Roma Tre' within the CRUI CARE Agreement