Task Planning on Stochastic Aisle Graphs for Precision Agriculture

Xinyue Kan, *Graduate Student Member, IEEE*, Thomas C. Thayer, *Graduate Student Member, IEEE*, Stefano Carpin, *Senior Member, IEEE*, and Konstantinos Karydis, *Member, IEEE*

***Abstract*—This work addresses task planning under uncertainty for precision agriculture applications whereby task costs are uncertain and the gain of completing a task is proportional to resource consumption (such as water consumption in precision irrigation). The goal is to complete all tasks while prioritizing those that are more urgent, and subject to diverse budget thresholds and stochastic costs for tasks. To describe agriculture-related environments that incorporate stochastic costs to complete tasks, a new Stochastic-Vertex-Cost Aisle Graph (SAG) is introduced. Then, a task allocation algorithm, termed Next-Best-Action Planning (NBA-P), is proposed. NBA-P utilizes the underlying structure enabled by SAG, and tackles the task planning problem by simultaneously determining the optimal tasks to perform and an optimal time to exit (i.e. return to a base station), at runtime. The proposed approach is tested with both simulated data and real-world experimental datasets collected in a commercial vineyard, in both single- and multi-robot scenarios. In all cases, NBA-P outperforms other evaluated methods in terms of return per visited vertex, wasted resources resulting from aborted tasks (i.e. when a budget threshold is exceeded), and total visited vertices.**

***Index Terms*—Planning, robotics and automation in agriculture and forestry, scheduling and coordination, task and motion planning.**

I. Introduction

AUTONOMOUS agricultural mobile robots become increasingly more capable for *persistent missions* like monitoring crop health [1] and sampling specimens [2] across extended spatio-temporal scales to enhance efficiency and productivity in precision agriculture [3]. An autonomous robot (or a team of them) needs to perform certain tasks in distinct locations of the environment subject to a specific budget [4] on the actions the robot can take (e.g., a maximum capacity of soil samples to carry [5]). During in-field operations, the actual costs to complete tasks can be uncertain whereas expected costs may be known. Also, some tasks can be more urgent than others, and have to be prioritized. It is often the case [3], [6], [7] that there exists some prior information about a required task (e.g., older measurements of soil moisture [8]) that can bias robot task assignment(s). Hence, it is necessary to develop approaches that utilize limited prior information to plan tasks with uncertain costs and priority level.

There exist two key challenges for efficient robot task allocation in precision agriculture. First, prior maps can indicate biases in task assignments, but may not be trustworthy. This is because conditions in the agricultural field can change rapidly [9], are dynamic [10], [11], and may be hard to predict ahead of time [12]. Second, as the budget is being depleted, the robot needs to periodically return to a base station (e.g., to drop collected samples and/or recharge). Addressing these two challenges simultaneously poses a two-layer intertwined decision making under uncertainty problem: *How to perform optimal sampling given an approximate prior map, and how to decide an optimal stopping time (i.e. to return to base) to avoid exceeding a given task capacity?* This letter introduces a new stochastic task allocation algorithm to balance optimal sampling and optimal stopping when task costs are uncertain.

A direct approach for persistent sampling (and/or monitoring) is to survey the entire space and perform the desired task(s) sequentially [13]–[15]. The main drawback is that the robot would then exhaustively visit all sampling locations without prioritizing those that would yield a higher gain or would be more time-critical. Orienteering [16]–[19] can address part of this drawback by determining paths that maximize the cumulative gain under a constant budget. The robot prioritizes visiting adjacent locations if they jointly yield higher gains than isolated high-gain locations, and provided that any budget constraints are not violated [16], [17]. However, this strategy can be insufficient for missions where some tasks are more urgent than others. For instance, several existing robot task allocation strategies, albeit for distinct application domains [20]–[23], typically consider a deadline [24] or user-defined importance levels. In precision agriculture, overhead imagery (e.g., thermal imaging) can help pinpoint locations that appear to be under water stress [9], in which case sampling leaves or soil in those areas should be prioritized. We formalize the notion of tasks with distinct urgency (e.g., a closer deadline or greater importance) by assigning a *priority level* [25], [26] to tasks.

Manuscript received October 15, 2020; accepted February 1, 2021. Date of publication February 25, 2021; date of current version March 23, 2021. This letter was recommended for publication by Associate Editor G. Garofalo and Editor Y. Choi upon evaluation of the reviewers' comments. This work was supported in part by NSF under Grants #IIS-1724341, #IIS-1901379, and #DGE-1633722 and in part by USDA-NIFA under Grants #2017-67021-25925 and #2021-67022-33453. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors' and do not necessarily reflect the views of the funding agencies. *(Corresponding author: Konstantinos Karydis.)*

Xinyue Kan and Konstantinos Karydis are with the Department of Electrical, and Computer Engineering, University of California, Riverside, 92521 USA (e-mail: xkan001@ucr.edu; kkarydis@ece.ucr.edu).

Thomas C. Thayer and Stefano Carpin are with the Department of Computer Science and Engineering, University of California, Merced, Merced, CA 95343 USA (e-mail: tthayer@ucmerced.edu; scarpin@ucmerced.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3062337, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3062337

Besides the task priority level, deciding a next task for a robot to complete is also dependent on available budget, which can be of multiple types. For instance, the number of locations that a robot can visit and sample from in one 'trip' is constrained by both the energy capacity to move between locations and the robot's sample payload capacity. Exceeding the energy budget can prevent the robot from returning to the base station to recharge and drop collected samples, whereas exceeding the sample payload capacity may cause potential robot and sample damage. Here we consider an *energy budget* for the robot moving between locations, and a *resource budget* linked to task execution. The two budgets are independent of each other, and both can be reset to their initial values when the robot returns to the base station. The actual amount of resources consumed to execute a task can differ from what is the expectation in practice. In fact, the actual amount of resources consumed for task execution is revealed only after the task has been completed. To model this, we consider the cost to complete a task to be a stochastic random variable that follows some known distribution. The cost to move between locations, however, is considered to be deterministic [8]. Specific details are given in the following.

This paper introduces a new stochastic task allocation approach, termed Next-Best-Action Planning (NBA-P), for task planning under uncertainty in precision agriculture. The paper also contributes a new Stochastic-Vertex-Cost Aisle Graph (SAG). SAG is an extension of the aisle graph [8], [27], which is often used to describe agriculture-related environments. The main novelty of SAG is that it can represent uncertain task costs. Using SAG, our proposed NBA-P algorithm simultaneously determines 1) how to optimally schedule which tasks to perform at run-time, and 2) when to optimally stop performing new tasks and return back to the base station also at run-time. NBA-P ensures that urgent tasks are prioritized subject to both energy and resource budgets. Further, it can be extended to multi-robot teams. We test in single- and multi-robot cases using both simulated data and 10 real-world datasets collected in a commercial vineyard at central California. In all cases, NBA-P achieves higher efficiency than naive lawnmower, informed lawnmower, and series Greedy Partial Row planners [28]–[30] in terms of more return per visited vertices, less resources wasted because of aborted tasks, and less total visited vertices.

II. Related Work

Aisle graphs [8], [27] model motion constraints emerging when robots navigate in structured environments like agricultural fields. Vertices denote task locations, and edges represent connections between locations. Any two rows connect to each other only via the two end vertices. Moving backwards is not allowed; if a robot enters a row, it will have to reach the row's end before moving to another row. In the original aisle graph [8], [27], vertices and edges are associated with known and constant reward and movement costs, respectively. Our extension, SAG, can also represent uncertain task costs.

Orienteering can tackle persistent sampling on aisle graphs. Orienteering is NP-hard, and thus greedy heuristics are often employed [8]. Recent efforts on stochastic orienteering associate stochastic costs to graph edges and propose a time-aware policy for a robot to adjust its path to avoid exceeding a certain budget [31]. However, addressing cases that involve uncertain task cost on vertices for aisle graphs remains open. NBA-P tackles the problem by simultaneously considering uncertain task costs on vertices and deterministic costs on edges.

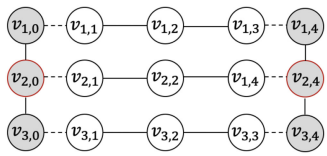

Fig. 1. SAG $A_s(3, 3+2)$ with 3 rows and 5 columns. Grey nodes are end vertices to connect rows. Vertices with red edges are base stations.

Optimal stopping [32] can be used to find the (optimal) criteria to terminate a process while incorporating uncertainty [33]. Often, data arrive in sequence, and irrevocable decision has to be made as to when the expected return is maximized [34]–[36]. Optimal stopping has been used in robotics applications like target tracking [37] and marine ecosystem monitoring [38]. However, no motion constraints, like those imposed by aisle graphs, apply to robot actions, and hence existing methods cannot be ported over to operations on aisle graphs. Paths planned with NBA-P fill the gap, as they directly apply to environments with motion constraints captured by aisle graphs.

Our method applies when: 1) motion constraints in the environment can be captured by a SAG; 2) the cost of completing tasks follow exponential distributions; and 3) the obtained gain by completing a task is proportional to the actual task cost.

III. Stochastic Task Allocation Problem Setup

We first define the *Stochastic-Vertex-Cost Aisle Graph (SAG)*, to incorporate uncertain task cost on vertices. Then, we present this paper's problem setup utilizing SAG.

A. The Stochastic-Vertex-Cost Aisle Graph (SAG)

We propose SAG to extend the original aisle graph [8], [27] to handle missions consisting of tasks with priority levels and stochastic execution costs. There are three main differences between SAG and the original aisle graphs. 1) SAG considers stochastic costs for task execution at vertices. 2) Vertices in SAG are associated with task priority levels. 3) The gain, which describes the benefit of completing a task, is proportional to the actual resource consumption if the task is fully completed. With more resource consumption, higher gain could be obtained, e.g., higher quality information during soil sampling process, or better field watering. Note that no gain will be obtained if 1) the resource budget is exceeded during task execution, and the task is aborted, or 2) a robot only passes through a vertex on its way without performing a task. In contrast, in the original aisle graph rewards are constant and can be collected immediately when passing through vertices.

Given a field that contains m rows and n columns (where n denotes the total number of possible sampling locations in each row), its SAG representation is an (undirected) graph $A_s(m, n+2) = (V, E)$ where V and E are the sets of vertices and edges, respectively. Note that in the graph representation we add two additional 'virtual' columns at indices $j = 0$ and $j = n+1$ that connect the m rows; virtual vertices carry no gain. An example of a $A_s(3, 5)$ graph is given in Fig. 1.

The set of edges E is built as follows:[1]

- Vertices $v_{i,j}$ with $i \in [1, m]$ and $j \in [1, n]$ have two edges, $e_{i,(j-1)^+}$ and e_{i,j^+}.
- Vertices $v_{i,j}$ with $i \in (1, m)$ and $j \in \{0, n+1\}$ have three edges: if $j = 0$ then $\{e_{i,0^+}, e_{(i-1)^+,0}, e_{i^+,0}\}$ and if $j = n+1$ then $\{e_{i,n^+}, e_{(i-1)^+,n+1}, e_{i^+,n+1}\}$.
- The four corner vertices are $v_{1,0}, v_{m,0}, v_{1,n+1}, v_{m,n+1}$, each of which has two edges.

Set S contains all priority levels in A_s. Let $c_v : V \to \mathbb{R}_{\geq 0}$ and $c_e : E \to \mathbb{R}_{\geq 0}$ be the costs for task execution at vertices and movement on edges, respectively. The actual resource consumption to complete a task at vertex $v \in V$ follows an exponential distribution, $c_v(v) \sim Exp(\bar{w}_s)$, where $\bar{w}_s$ is the mean cost of all tasks with priority level $s \in S$. The actual task cost is not known before task completion, and is independent between tasks at different locations. The cost of movement on edges is a known constant. Function $f : V \to S$ returns the priority level of a vertex, and $f(v_{i,j}) = 0, v_{i,j} \in V$ indicates no tasks at a vertex.[2] Condition $f_{(i_1,j_1)} < f_{(i_2,j_2)}$ implies that the task at v_{i_2,j_2} is more urgent than the task at v_{i_1,j_1}. In other words, if the same amount of resources is consumed at v_{i_1,j_1} and v_{i_2,j_2}, higher gain is obtained at v_{i_2,j_2}. Once a task is completed, its priority level is set to 0.

Let $r : V \to \mathbb{R}_{\geq 0}$ be the actual gain obtained when completing a task. Function $\mu : S \to \mathbb{R}_{>0}$ maps each priority level to a deterministic positive value, which indicates the gain-to-cost ratio of completing a task of given priority level. Then, $r_{(i_1,j_1)} = \mu(f_{(i_1,j_1)})c_{v(i_1,j_1)}$, and $\mu(f_{(i_1,j_1)}) < \mu(f_{(i_2,j_2)})$ if $f_{(i_1,j_1)} < f_{(i_2,j_2)}$ for $v_{i_1,j_1}, v_{i_2,j_2} \in V$. Vertices $v_{i,j}$ with $i \in [1, m]$ and $j = n+1$ are virtual nodes to connect rows, hence $f_{(i,j)} = 0$, $c_{v(i,j)} = 0, r_{(i,j)} = 0$. Edges $e_{i,0^+}$ and e_{i,n^+} with $i \in [1, m]$ has $c_e(e_{i,0^+}) = c_e(e_{i,n^+}) = 0$.

Priority levels can be user-defined or estimated via any prior environment maps. The latter can be determined based on collected data, e.g., difference between ideal and sampled soil moisture levels [8]. However, prior information may be approximate and thus lead to suboptimality if directly set as priority levels for vertices. A way to assign priority levels from prior information is to set thresholds so that data within a range yield the same priority level. Only same types of tasks with same expected cost can be set at same priority level.

B. Stochastic Task Allocation on SAGs

A mission on SAG $A_s(m, n+2) = (V, E)$ comprises tasks located at $v \in V_T \subset V$. Given energy budget for moving along edges and resource budget for executing tasks on vertices, to complete all tasks in V_T so that:

- C1: Tasks are prioritized according to priority level.
- C2: The number of tasks being aborted because of exceeding the resource budget (at run-time) are minimized.

C1 enforces the time-critical decision making, whereas C2 ensures efficiency of mission completion. When a task is aborted, no gain is obtained and both consumed resources and energy spent moving to that vertex are wasted. Aborting tasks will also cause delays on mission completion time. To avoid exceeding the resource budget at run-time, the robot thus needs to determine an optimal stopping time. Its next action should be to either 1) perform another feasible task of the highest possible priority level (which we describe how to set next), or 2) stop performing tasks and return to the base station. Since the actual cost is unknown before completing a task, the next action and corresponding paths are determined in an adaptive manner based on remaining budget at run-time.

IV. Proposed Task Planning Algorithm

Our proposed Next-Best-Action Planning (NBA-P) approach balances sampling feasible vertices on SAG and determining when it is preferable to exit (i.e. return to base station) based on remaining resource and energy budgets. When sampling feasible vertices, we use a three-phase approach. Phase 1: sample feasible vertices subject to resource budget; Phase 2: sample feasible vertices from phase 1 subject to energy budget; Phase 3: select a row to proceed and plan corresponding paths. When sampling in phase 1, we start from the highest priority level that currently exists. If either phase 1 or phase 2 returns no feasible vertex, we decrease the examined priority level until either feasible vertices are found, or the examined priority level reaches 0, in which case it is optimal to exit. This strategy ensures that tasks with higher priority level are prioritized when possible.

A. Phase 1: Feasible Vertices Subject to Resource Budget

To tackle the stochastic task cost, we formulate the next-task selection subject to resource budget as an optimal stopping problem. We employ a one-stage-look-ahead rule: if it is better to *return to base station directly* than to *perform one more task of any priority level then return*, then return at current time. In this phase, we do not need to consider the actual robot position. Let p and q be the remaining resource budget and the total gain in the current 'trip' (i.e. operation since last visit to a base station), respectively.[3] If a task of priority level $s \in S$ consumes x amount of resources, the return is $\mu(s)x$. Then, in a dynamic programming framework, with (p, q) the state, the expected return function, $\Phi(p, q)$, is

$$\Phi(p,q) = \begin{cases} \max\limits_{s \in S} \left\{ \int_0^p \lambda_s e^{-\lambda_s x} \Phi(p-x, q+\mu(s)x)dx \right\}, & \text{if p} > 0 \\ q, \text{ otherwise,} \end{cases} \tag{1}$$

where $\lambda_s = \frac{1}{\bar{w}_s}$. When $p > 0$ (i.e. some resource is available), a task of priority level $s \in S$ which maximizes the return is selected. Otherwise, no task can be completed and the total return remains the same as q.

1) Single Priority Level for All Tasks: We start with the case that all tasks in the mission have the same priority level, $|S| = 1$. In this case, we only need to determine the optimal time to exit. According to (1), for $s \in S$, the state $(p,' q')$ is on the optimal stopping boundary if

$$q' = \int_0^{p'} \lambda_s e^{-\lambda_s x}(q' + \mu(s)x)dx, \tag{2}$$

[1]For clarity and completeness, we follow and partially adapt the definition of the original aisle graphs from [8], [27].

[2]For clarity, we will henceforth write $f(v_{i,j})$ as $f_{(i,j)}$. Any other functions that take vertices as input will be shortened similarly.

[3]Gain q are set to 0 when the robot resets at the base station.

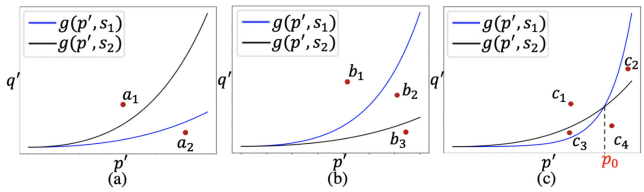

Fig. 2. Illustrations of cases described in (a) Lemma 1, (b) Condition 1 and (c) Condition 2. Red points visualize sample states (p, q). In (a)-(c), tasks of priority levels s_1 or s_2 are feasible if the current state (p, q) is below the curves $g(p,' s_1)$ or $g(p,' s_2)$, respectively.

since continuing to perform another task will not result in higher expected return. Hence, the robot should exit if the current state (p, q) satisfies $p < p'$ and $q > q'$, i.e. all tasks are infeasible. Solving (2) leads to

$$q' = \frac{\mu(s)}{\lambda_s}(e^{\lambda_s p'} - 1 - \lambda_s p'). \tag{3}$$

Defining function $g : (\mathbb{R}_{\geq 0}, S) \to \mathbb{R}_{\geq 0}, (p,' s) \mapsto q'$ based on (3) represents the optimal stopping boundary curve for a given priority level. Thus, it is optimal to exit at state $(p,' q')$ when $q' \geq g(p,' s)$ given a priority level $s \in A_s$.

Definition 1: A task of priority level s is *feasible* for the current state (p, q), if (p, q) lies below the optimal stopping boundary curve $g(p,' s)$ (Fig. 2).

2) Multiple Priority Levels Across Tasks: If $|S| > 1$, the robot determines the candidates with highest possible priority level allowed by the remaining budget. The optimal strategy is to examine the feasibility to perform a task of priority level $s = \max(S)$, and then decrease s until a feasible task is found. If no feasible task exists until $s = 0$, then the optimal decision is to return back to the base station.

When multiple priority levels exist, it is not always true that tasks with higher priority levels must be performed before any lower priority rank tasks. To maximize the expected return in one 'trip' (i.e. between two times that a robot visits the base station), when the remaining resource budget is not enough for high priority level tasks, a task with lower priority level can potentially be selected to be performed next. However, in some scenarios, a lower priority task will never be selected prior to a higher priority task.

Lemma 1: At state (p, q), given that tasks with priority level $s \in S$ are infeasible, then all tasks with $s' \in S$ and $s' < s$ must be infeasible if $\bar{w}_{s'} \geq \bar{w}_s$.

Proof: Let $s_1, s_2 \in S : 1 \leq s_1 < s_2 \leq max(S)$ and $\mu(s_1) < \mu(s_2)$. The mean costs of s_1 and s_2 tasks are $\bar{w}_{s_1}$ and $\bar{w}_{s_2}$, respectively. If $\bar{w}_{s_1} \geq \bar{w}_{s_2}$, for any $p > 0$, $g(p, s_2) > g(p, s_1)$. Hence, if a s_2 priority level task is infeasible at state (p, q), i.e. $q \geq g(p, s_2)$, then $q \geq g(p, s_2) > g(p, s_1)$, and s_1 tasks are infeasible too (Fig. 2(a)). ■

On the other hand, as shown in Fig. 2(b), (c), for $s_1, s_2 \in S$ such that $1 \leq s_1 < s_2 \leq max(S)$, if $\bar{w}_{s_1} < \bar{w}_{s_2}$, the relationship between boundary curves $g(p, s_1)$ and $g(p, s_2)$ is either Condition 1 (Fig. 2(b)) or Condition 2 (Fig. 2(c)).

- Condition 1 :

$$g(p, s_2) < g(p, s_1), \forall p > 0, \tag{4}$$

Algorithm 1: SampleQone$((p, q), s)$.

```
procedure DETERMINE Q_1 AT STATE (p, q)
  s ← min(s, max(S)), Q_1 ← ∅
  while s > 0 And Q_1 = ∅ do
    if q < g(p, s) then
      for v ∈ V_T do
        if f(v) = s then Q_1 ← Q_1 ∪ {v}
        end if
      end for
    else
      while w_{s−1} ≥ w_s do s ← s − 1
      end while
      s ← s − 1
    end if
  end while
  return Q_1, s
end Procedure
```

- Condition 2: $\exists p_0 > 0$, such that

$$g(p, s_2) \begin{cases} \geq g(p, s_1), & \text{if } 0 < p > p_0 \\ < g(p, s_1), & \text{if } p > p_0 \end{cases} \tag{5}$$

For Condition 1, a state (p, q) above the curve $g(p, s_2)$ can be still below the curve $g(p, s_1)$, e.g., point b_2 in Fig. 2(b). In this case, s_1 tasks should be performed next even if there still exist s_2 tasks. For Condition 2, when $p > p_0$, the situation is the same as described above for Condition 1. When $0 < p \leq p_0$ we reduce to the conditions of Lemma 1, in which case given that s_2 tasks are infeasible, s_1 tasks must be infeasible (an example is point c_1 in Fig. 2(c)).

Let Q_1 be the set containing all feasible vertices subject to a given resource budget. We propose Algorithm 1 to determine Q_1 at a state (p, q). If $Q_1 = \emptyset$, the robot returns to the base station. Otherwise, all vertices in Q_1 will continue to be examined in Phase 2 subject to a given energy budget.

B. Phase 2: Feasible Vertices Subject to Energy Budget

From Q_1, we continue sampling vertices that satisfy the energy budget constraint. Suppose a robot is at vertex $v_{i_c,j_c} \in V$, and a vertex $v_{i,j} \in Q_1$ is a candidate to be examined. Without loss of generality, suppose two base stations are located on row i_d, each at one of the end vertices $v_{i_d,0}$ and $v_{i_d,n+1}$. The robot can reset at either one. Vertex $v_{i',j'}$ is feasible if the current remaining energy budget T allows the robot to move to $v_{i',j'}$ then to any base station. Let $t_\alpha(i')$ be the cost to move from v_{i_c,j_c} to an end node–either on column 0 or $n + 1$ depending on the robot's moving direction in current row i' (recall backward motion is not allowed). Let $t_\beta(i')$ be the cost to move between two end vertices $v_{i',0}$ and $v_{i',n+1}$ in row i', and t_γ be the cost to move from the end vertex on row i' closest to the robot along its direction of motion to the closest base station. Then, $v_{i',j'}$ is feasible if [4]

$$T \geq t_\alpha(i') + t_\beta(i') + t_\gamma(i'). \tag{6}$$

If $v_{i',j'}$ can be reached, all vertices on row i' must be reachable, since $t_\alpha(i') + t_\beta(i') + t_\gamma(i')$ only depends on row i'. Costs t_β

[4]Expressions to compute t_α, t_β and t_γ are given in the Appendix.

and t_γ are fixed for each row and can be precomputed prior to deployment. Cost t_α is computed at run-time. The set containing all vertices that satisfy both budgets is $Q_2 = \{v_{i;j'} \in Q_1 | t_\alpha(i') + t_\beta(i') + t_\gamma(i') \leq T\}$.

C. Phase 3 and Proposed Algorithm

Phase 3 can be reached if $Q_2 \neq \emptyset$. Note that all tasks at vertices in Q_2 have the same priority level s and hence the same expected cost $\bar{w}_s$. Therefore, sampling the next vertex turns into selecting a row i which consists of one or more tasks of priority level s. Then, the robot will perform the first encountered feasible task while moving along row i.

Suppose the robot is currently at v_{i_c,j_c} with state (p, q). The row $i \in [1, m]$ is selected such that $\forall i' \in [1, m], i' \neq i_c$,

$$\min\{|Q_2(i)|, \lfloor \frac{q}{\bar{w}_s} \rfloor\} > \min\{|Q_2(i')|, \lfloor \frac{q}{\bar{w}_s} \rfloor\}, \tag{7}$$

$$t_\alpha(i) \leq t_\alpha(i'), \tag{8}$$

where $Q_2(i) = \{v_{i,j'} \in Q_2 | i' = i\}$, and $\bar{w}_s$ is the mean cost of feasible tasks. By (7), the robot is expected to complete more tasks in row i than any other row. Thus, row i should be the row that contains the largest number of feasible tasks permitted by remaining budget q, according to the expected cost $\bar{w}_s$. If multiple rows return a tie, then the row closest to the robot's current position will be selected as per (8).

The proposed Next-Best-Action Planning (NBA-P) approach is formalized in Algorithm 2. NBA-P can be extended to apply to multi-robot teams by sequentially determining the next best action for each robot. In multi-robot implementation, each robot runs NBA-P independently and in parallel, and exchanges information only about the row it currently occupies. For each robot, Q_2 has to be modified by removing all vertices in those rows that are occupied by other robots. Note that similar to [16], multiple robots can travel simultaneously along the vertical columns 0 and $n + 1$, since space on the boundary of a field is typically much larger.

Algorithm 2: NBA-P.

```
Procedure Determine next action at state (p, q)
    s ← max(S),
    Q1 ← ∅, Q2 ← ∅
    While Q2 = ∅
      If s=0 exit, return to base station
      End If
      Q1, s ← SampleQone((p, q), s)
      If Q1 = ∅ exit, return to base station
      Else
        obtain Q2 from Eq. (6), s ← s − 1
      End If
    End While
    continue with Phase 3
End Procedure
```

V. Results

To study the efficiency and effectiveness of the proposed approach, we test with 1) simulated data in a 2-robot scenario, and 2) data collected from a real-world vineyard in 1- and 5-robot scenarios. Testing with simulated data enables parameter tuning so as to study the properties of NBA-P, whereas testing with real-world data reveals the spatial pattern of real tasks that exist in agricultural fields. In both cases, NBA-P is compared against lawnmower planner [28], which is often seen in agriculture-related applications [29], [30]. In naive lawnmower (N-LM), a robot follows meandering paths to survey rows in sequence. When no budget constraint is considered, and when departing from a corner in a square environment, lawnmower will generate the shortest path to survey the entire field in the sense that each vertex is visited only once. In experiments using real-world datasets, NBA-P is also compared against informed lawnmower (I-LM) and Series Greedy Partial Row (S-GPR) [16]. I-LM attempts to complete a task if the remaining budget is greater than the expected cost of a task. S-GPR is modified to use both energy and resource budgets. In multi-robot cases, each robot runs N-LM, I-LM, and NBA-P independently and in parallel to each other; in S-GPR robots plan their trajectories sequentially. All vertices in occupied rows turn infeasible for other robots so that each row has one robot performing tasks.

A. Testing With Simulated Data and Parametric Study

We consider a simulated environment $A_s(20, 17)$ of 20 rows and 17 columns (including the two virtual columns). Base stations are located at $v_{10,0}$ and $v_{10,16}$. The cost to move on each edge is 1. Consider two robots deployed from base station $v_{10,0}$ to complete all tasks. Each robot departs with energy budget 80, and resource budget 40.[5] A vertex is considered "visited" if the robot stops at the vertex and attempts to perform a task, regardless whether the task is ultimately completed or aborted. If the resource budget is exceeded before task completion, the task will be aborted, the resources already consumed for this task are considered to be "wasted," and no gain will be obtained. The total gain will be the sum of the actual gain, which is proportional to the actual task cost, at all task-completed vertices. Two cases are studied, 1) $S = \{1\}$, i.e. all tasks have equal priority level, and 2) $S = \{1, 2\}$, i.e. two priority levels exist, hence tasks with $s = 2$ will be prioritized. In case 1, $\mu(1) = 1, \bar{w}_1 = 2$; and in case 2, $\mu(1) = 1, \mu(2) = 2, \bar{w}_1 = 1.5, \bar{w}_2 = 2$. For each case study, 10 trials are conducted. In each trial, 225 tasks are randomly assigned to 225 vertices in A_s, with randomly generated task location and actual task cost. In each trial, the proposed method and the N-LM method are tested on the same simulated environment.

Fig. 3 illustrates shows from a sample trial when $|S| = 1$ (top) and $|S| = 2$ (bottom). Fig.s 3(a) and (c) show the percentage of obtained gain over ground truth total gain as a function of visited vertices (shortened as r/v ratio). Fig.s 3(b) and (d) show the total wasted resources because of aborted tasks as a function of visited vertices (shortened as w/v ratio). Total wasted resources are the sum of resource consumption for all aborted tasks. Total gain, total wasted resources and visited vertices correspond to the sum of those values from both robots. Results suggest that all tasks are completed, and the robots return to the base station.

[5]Values for both budgets are selected randomly for evaluation purposes. Expected task costs are selected to achieve a budget over expected cost ratio of 20, which helps reveal general patterns of the method.

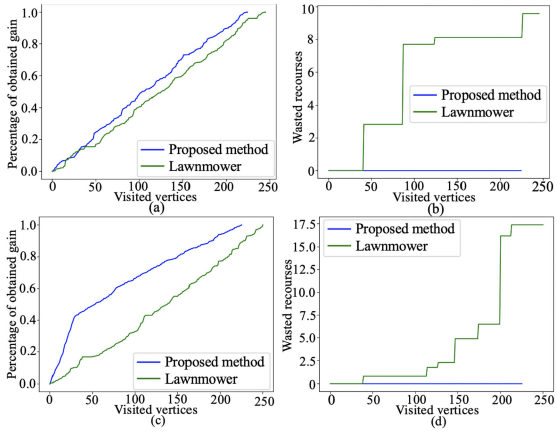

Fig. 3. Example of (a) percentage of obtained gain and (b) wasted resources over visited vertices when $|S| = 1$. Panels (c) and (d) contain the same information when $|S| = 2$.

TABLE I
RESULTS FOR SIMULATED DATA OVER 10 TRIALS

		r/v ratio (10^{-3})	w/v ratio (10^{-2})	vertices visited
$\|S\| = 1$	NBA-P	$\mathbf{4.42 \pm 0.02}$	$\mathbf{1.12 \pm 1.08}$	$\mathbf{226.2 \pm 1.1}$
	N-LM	3.46 ± 0.18	9.13 ± 1.74	289.4 ± 14.9
$\|S\| = 2$	NBA-P	$\mathbf{4.44 \pm 0.02}$	$\mathbf{0.33 \pm 0.73}$	$\mathbf{225.4 \pm 0.7}$
	N-LM	3.76 ± 0.1	4.5 ± 1.63	266.2 ± 7.0

Table I contains the mean and one standard deviation of r/v ratio, w/v ratio, and total visited vertices over 10 trials. Larger r/v suggests higher efficiency since more gain is obtained by visiting the same number of vertices, i.e. same number of attempts to execute tasks. Lower w/v indicates lower rates of aborted tasks, i.e. less resources are wasted by visiting the same number of vertices. Higher r/v, lower w/v, and less total visited vertices are desired, and these conditions together indicate higher overall effectiveness.

Results in Fig. 3 and Table I suggest that, in both cases, NBA-P achieves higher r/v ratio, lower w/v ratio, and less total visited vertices than N-LM. When $|S| = 1$, since all tasks have the same priority level, the higher r/v ratio of NBA-P is mainly due to the optimal stopping strategy that helps prevent aborting tasks. When $|S| = 2$, the higher r/v is due to both the optimal stopping and priority-driven strategies. This can be observed by the steep slope at the beginning of the curve of our method in Fig. 3(c), during which time tasks with priority level 2 are prioritized. The high rate of tasks being aborted in N-LM explains why the total visited vertices for N-LM are more than for NBA-P. For N-LM, the robot will attempt to perform a task if there is still remaining resource budget. However, if the budget is exceeded during task execution, the task will be aborted and the vertex needs to be re-visited. Setting multiple priorities may be useful but needs to be carefully tuned as too many priority levels can make the process inefficient in practice, forcing the robot to move across the field to reach the tasks of the next highest priority level.

Reducing the rate of aborted tasks can increase efficiency. We continue to study the influence of μ and $\bar{w}_s$ to the rate of aborted tasks, in which case the energy budget does not need to be considered. Starting with $|S| = 1$, and assuming there exist

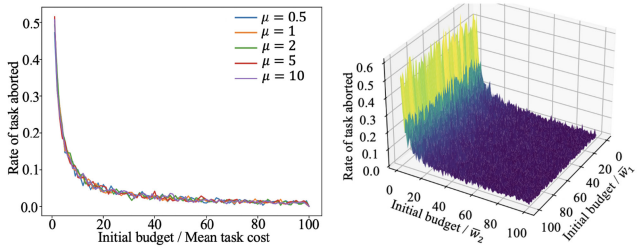

Fig. 4. (Left) Rate of aborted tasks over the ratio of budget/$\bar{w}_1$ when $|S| = 1$. (Right) Rate of aborted tasks with respect to the ratio of budget/$\bar{w}_1$ and budge=t/$\bar{w}_2$ for the $|S| = 2$ case.

infinite tasks, we need to determine the optimal time to stop performing more tasks. Fig. 4 (left) shows the relation between aborted tasks (ratio of occurrences over 1000 trials) and the ratio of initial budget over mean task cost $\bar{w}_1$, for different values of μ. Results suggest that aborting a task is barely influenced by the value of μ. If the initial budget is close to the mean task cost, the rate of aborted tasks can be as high as 50%, and the optimal stopping rule is less effective. If the initial budget is more than 50 times the mean task cost, the rate of aborted tasks reaches 0.

Given that μ barely affects the rate of aborted tasks (Fig. 4 (left)), we examine in the case that $|S| = 2$ the influence of the ratio between initial budget and task mean cost for each priority level. We assume there exist infinite tasks of priority level 1 and 2. The goal is to determine if it is better to select another task of priority level 1 or 2 to perform, or to stop. Fig. 4 (right) suggests that, regardless of the relation between $\bar{w}_1$ and $\bar{w}_2$, the rate of aborted tasks is more influenced by the ratio of initial budget over $\bar{w}_2$. This is intuitive since tasks of priority level 2 are prioritized over tasks of priority level 1. Since more $s = 2$ tasks are performed if possible, it escalates its influence to the rate of aborting tasks. Thus, when higher priority tasks exist, the initial budget can be set by considering expected cost of high priority tasks, as the expected cost of low priority tasks do not have much impact when energy is sufficient. The proposed method is more suitable when the mean cost of tasks is small enough compared to the initial (resource) budget.

B. *Testing With Real-World Field Data*

The real-world datasets used here contain soil moisture values collected in a commercial vineyard located in central California. The structure of the vineyard imposes motion constraints to ground robots moving therein. Irrigation lines are attached to metallic wires at about 12 in from the ground and running parallel to the trellis. Thus, to move from one row to another (even if adjacent), the robot must first exit the row from either end (based on its direction), and then re-enter at the desired row. Samples were collected on a regular grid with 275 rows and 214 columns uniformly covering the field. Sampling locations were computed offline, and data were collected with a Campbell H2SP soil moisture sensor.

Suppose autonomous ground mobile robots are deployed to water plants in the vineyard. Vertices with moisture values less than a desired level are considered to contain a task (of precise watering). The ground truth task cost at any vertex is the moisture difference between collected moisture values and the desired. An

TABLE II
RESULTS FOR 10 FIELD EXPERIMENTAL DATASETS IN 1- AND 5-ROBOT SCENARIOS

robot number	data idx / total tasks		1 39528	2 44607	3 58845	4 38190	5 33489	6 24075	7 12203	8 58553	9 58551	10 20345
1	r/v (10^{-5})	NBA-P	**2.53**	**2.24**	**1.70**	**2.62**	**2.99**	**4.15**	**8.19**	**1.71**	**1.71**	**4.92**
		N-LM	1.72	1.23	0.55	1.23	1.56	2.17	6.68	0.59	0.57	3.72
		I-LM	2.53	2.24	1.70	2.61	2.98	4.14	8.18	1.70	1.70	4.91
		S-GPR	2.52	2.22	1.67	2.59	2.95	4.10	8.15	1.68	1.68	4.89
	w/v (10^{-3})	NBA-P	**0.15**	**0**	**0**	**0.61**	**0**	**0**	**1.61**	**0**	**0**	**1.44**
		N-LM	5.88	11.9	23.42	22.07	16.96	21.37	6.66	19.45	24.53	4.91
		I-LM	4.62	9.98	16.77	28.87	19.10	23.47	7.31	17.23	27.18	4.00
		S-GPR	9.12	22.44	70.44	49.03	33.47	41.83	10.94	58.66	76.18	7.70
	visited vertices	NBA-P	**39529**	**44607**	**58845**	**38192**	**33489**	**24075**	**12206**	**58553**	**58551**	**20345**
		N-LM	58220	81277	181553	81083	64178	46144	14967	168360	176174	26914
		I-LM	39585	44703	58966	38349	33600	24159	12223	58689	58734	20372
		S-GPR	39731	45018	59933	38672	33847	24363	12263	59504	59587	20433
5	r/v (10^{-5})	NBA-P	**2.53**	**2.24**	**1.70**	**2.62**	**2.99**	**4.15**	**8.19**	**1.71**	**1.71**	**4.91**
		N-LM	1.77	1.3	0.62	1.37	1.69	2.33	6.81	6.52	6.03	3.81
		I-LM	2.53	2.24	1.70	2.61	2.98	4.14	8.18	1.70	1.70	4.91
		S-GPR	2.52	2.22	1.67	2.59	2.95	4.10	8.15	1.68	1.68	4.89
	w/v (10^{-3})	NBA-P	**0**	**0**	**0**	**0.92**	**0**	**0**	**2.67**	**0**	**0**	**0.59**
		N-LM	5.97	12.00	24.72	24.67	18.66	22.62	6.23	21.18	27.05	4.72
		I-LM	5.81	8.37	14.78	27.35	18.47	23.58	4.94	14.66	29.46	3.61
		S-GPR	9.08	22.66	70.37	49.03	33.09	41.31	10.26	58.66	76.18	7.79
	visited vertices	NBA-P	**39528**	**44607**	**58845**	**38193**	**33489**	**24075**	**12209**	**58553**	**58551**	**20348**
		N-LM	56543	77210	162420	73179	59253	42923	14688	153425	165869	26233
		I-LM	39599	44687	58952	38335	33596	24160	12218	58670	58753	20369
		S-GPR	39729	45020	59933	38672	33847	24363	12263	59504	59587	20433

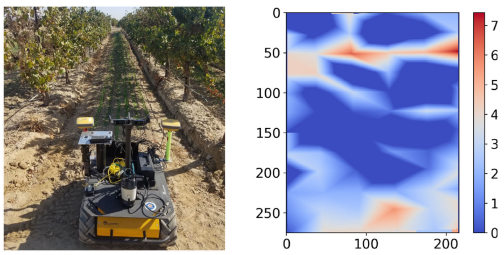

Fig. 5. (Left) Instance of a wheeled mobile robot performing soil moisture measurements in a commercial vineyard. (Right) Sample ground truth cost (i.e. the moisture difference between collected moisture values and a desired level) for one of the field experimental datasets used here. Low-moisture (dry) locations are indicated by higher differences. In these areas, more water (the resource in this case) needs to be consumed to reach a desired moisture level, which is equivalent to a higher cost. Discretely-sampled values where mapped to a continuous contour illustrated here using the kriging interpolation algorithm. (However, we use the discrete values directly on the SAG representation of the environment utilized by our algorithm.).

example in shown in Fig. 5. The robots' decision is constrained by the resource budget of total water carrying capacity, and the energy budget to move between locations. All tasks are considered to have the same priority level ($|S| = 1$). The location of tasks and the mean cost of all tasks (averaged real costs of all tasks using ground truth) are available to the robot(s) prior to departure, whereas the actual cost for each task is unknown to the robot(s) before task completion. Without loss of generality, we consider the movement cost on edges to be 1 (all water emitters are located within the same interval distance). We test in 1- and 5-robot cases; robots depart with energy budget 800 and resource budget 400. Base stations are located at $v_{137,0}$ and $v_{137\,215}$.

Table II shows results for 10 real-world datasets in 1- and 5-robot scenarios. The 10 datasets were collected during a timespan of 5 months, hence task locations and costs differ among datasets. Results suggest that, in all cases, our proposed NBA-P algorithm achieves higher r/v ratio, lower w/v ratio, and less total visited vertices than N-LM. Even though I-LM and S-GPR achieve similar r/v ratio and total visited vertices as NBA-P, the w/v ratio is much higher compared to NBA-P. Results attempts only if the expected task cost is less than the remaining budget yet it fails to consider the uncertainties in actual resource consumption, which can be much higher than the expected value. In addition, the high actual cost may cause multiple failed attempts at the same position. Thus, the total waste of resource can be significant, evident by the averaged total waste over 10 datasets for 1-robot cases being 5, 1725, 671 and 1796 for NBA-P, N-LM, I-LM and S-GPR, respectively. That is, NBA-P is able to handle uncertain task costs, and requires less total resources to complete the same amount of tasks as compared to other methods. Importantly, no tasks are aborted in 13 out of 20 cases using NBA-P, where each case contains up to 60 000 tasks. For datasets 3, 8, and 9, N-LM in fact visits three times more vertices than the proposed method. The total path lengths for evaluated methods differ within around 2%-range, and NBA-P achieves the shortest path for datasets 6 and 10. That is, NBA-P plans paths of similar length as lawnmower methods.

In all, testing with experimental data validates the efficacy and efficiency of our proposed method, and demonstrates preliminary feasibility that it can scale both in terms of the size of the environment and the number of robots in the team.

VI. CONCLUSION

Contributions and Key Findings. The paper contributes to stochastic task allocation in precision agriculture. Given resource and energy budgets, our NBA-P algorithm returns the best action on a stochastic aisle graph (SAG) by simultaneously determining optimal sampling locations and optimal stopping times to return to a base station, all at run-time. The proposed algorithm is tested using both simulated data for a 2-robot scenario and agricultural field experimental datasets for 1- and 5-robot scenarios. Results suggest that, by applying NBA-P, all tasks are completed, and tasks with high priority levels are prioritized when possible. The rate of aborted tasks is minimal when the initial resource budget is more than 50 times the mean task costs. NBA-P outperforms N-LM, I-LM and S-GRP methods in all

simulated and real-world datasets, in terms of more gain per vertex visited, fewer tasks being aborted, and less total visited vertices to complete the same number of tasks. In testing with real-world datasets, our method has no tasks aborted in 13 out of 20 cases with up to 60 000 tasks in each case. Further, N-LM visits up to 3 times more vertices than NBA-P to complete same number of tasks, which leads to a significant waste of resources.

Directions for Future Work. At its current form, NBA-P is not suitable for scenarios where the task and movement costs are correlated. Further, the overall paths using NBA-P can be longer than those derived via the lawnmower method, especially when multiple priority levels exist and tasks of different priority level are intertwined. Future directions of research include 1) application of the proposed algorithm to physical robots in the field, and 2) study of the scenario that considers correlated cost for movement and task execution.

APPENDIX

Let *hl* and *hr* represent the robot moving toward column 0 and $j+1$, respectively. Suppose $i_{11} = min(i_c, i')$, $i_{12} = max(i_c, i')$, $i_{21} = min(i,' i_d)$, and $i_{22} = max(i,' i_d)$. Then,

$$t_\alpha(i') = \begin{cases} \sum_{j=j_c}^{n-1} c_e(e_{i_c,j^+}) + \sum_{i=i_{11}}^{i_{12}-1} c_e(e_{i^+,n+1}), \text{if } hl, \\ \sum_{j=1}^{j_c-1} c_e(e_{i_c,j^+}) + \sum_{p=i_{11}}^{i_{12}-1} c_v(e_{i^+,0}), \text{if } hr. \end{cases}$$

$$t_\beta(i') = \sum_{j=1}^{n-1} c_e(e_{i,'j^+}).$$

$$t_\gamma(i') = \begin{cases} \sum_{i=i_{21}}^{i_{22}-1} c_e(e_{i^+,0}), \text{if } hl, \\ \sum_{i=i_{21}}^{i_{22}-1} c_e(e_{i^+,n+1}), \text{if } hr. \end{cases}$$

REFERENCES

[1] M. Bietresato *et al.*, "A tracked mobile robotic lab for monitoring the plants volume and health," in *Proc. IEEE/ASME Int. Conf. Mechatronic Embedded Syst. Apl*., 2016, pp. 1–6.

[2] F. A. Auat 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.

[3] J. J. Roldán *et al.*, "Robots in agriculture: State of art and practical experiences,"*Service Robots*, 2018.

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

[5] E. Vaeljaots *et al.* "Soil sampling automation case-study using unmanned ground vehicle," *Eng. Rural Dev.*, vol. 17, pp. 982–987, 2018.

[6] A. Pretto *et al.*, "Building an aerial-ground robotics system for precision farming," 2019, arXiv:1911.03098.

[7] S. Bonadies, A. Lefcourt, and S. A. Gadsden, "A survey of unmanned ground vehicles with applications to agricultural and environmental sensing," in *Autonomous Air Ground Sensing Syst. Argi. Opt. Phenotyping*, vol. 9866, 2016, Art. no. 98660.

[8] T. C. Thayer, S. Vougioukas, K. Goldberg, and S. Carpin, "Routing algorithms for robot assisted precision irrigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 2221–2228.

[9] J. Gago *et al.*, "UAVs challenge to assess water stress for sustainable agriculture," *Agri. Water Mgmt*, vol. 153, pp. 9–19, 2015.

[10] Z. Zheng *et al.*, "Spatiotemporal changes in soil salinity in a drip-irrigated field," *Geoderma*, vol. 149, no. 3-4, pp. 243–248, 2009.

[11] H. G. Jones, *Plants and Microclimate: A Quantitative Approach to Environmental Plant Physiology*. Cambridge, U.K.: Cambridge univ. Press, 2013.

[12] J. Kelly *et al.*, "Challenges and best practices for deriving temperature data from an uncalibrated UAV thermal infrared camera," *Remote Sens.*, vol. 11, no. 5, p. 567, 2019.

[13] H. Viet, V.-H. Dang, M. Laskar, and T. Chung, "Ba: An online complete coverage algorithm for cleaning robots," *Appl. Intell.*, vol. 39, 2013.

[14] X. Kan, H. Teng, and K. Karydis, "Multi-robot field exploration in hexdecomposed environments for dubins vehicles," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2019, pp. 449–455.

[15] X. Kan, H. Teng, and K. Karydis, "Online exploration and coverage planning in unknown obstacle-cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5969–5976, Oct. 2020.

[16] T. C. Thayer, S. Vougioukas, K. Goldberg, and S. Carpin, "Multi-robot routing algorithms for robots operating in vineyards," in *Proc. IEEE Int. Conf. Autom. Sci. Engr.*, 2018, pp. 14–21.

[17] T. C. Thayer, S. Vougioukas, K. Goldberg, and S. Carpin, "Bi-objective routing for robotic irrigation and sampling in vineyards," in *Proc. IEEE Int. Conf. Autom. Sci. Engr*., 2019, pp. 1481–1488.

[18] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *Eur. J. Opr. Res.*, vol. 255, no. 2, pp. 315–332, 2016.

[19] 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, 2016.

[20] F. Auat Cheein, M. Torres-Torriti, N. B. Hopfenblatt, Á J. Prado, and D. Calabi, "Agricultural service unit motion planning under harvesting scheduling and terrain constraints," *J. Field Robot.*, vol. 34, no. 8, pp. 1531–1542, 2017.

[21] P. R. Wurman, R. D'Andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *AI magazine*, vol. 29, no. 1, pp. 9–9, 2008.

[22] M. Veloso, J. Biswas, B. Coltin, and S. Rosenthal, "Cobots: Robust symbiotic autonomous mobile service robots," in *Proc. Int. Joint Conf. Artif. Intell*., 2015, pp. 4423–4429.

[23] Y. Khaluf, S. Vanhee, and P. Simoens, "Local ant system for allocating robot swarms to time-constrained tasks," *J. Comput. Sci.*, vol. 31, pp. 33–44, 2019.

[24] H. Ma, G. Wagner, A. Felner, J. Li, T. K. S. Kumar, and S. Koenig, "Multi-agent path finding with deadlines," *CoRR*, vol. abs/1806.04216, 2018.

[25] L. B. Becker, E. Nett, S. Schemmer, and M. Gergeleit, "Robust scheduling in team-robotics," *J. Syst. Softw.*, vol. 77, no. 1, pp. 3–16, 2005.

[26] A. Maoudj, B. Bouzouia, A. Hentout, A. Kouider, and R. Toumi, "Distributed multi-agent approach based on priority rules and genetic algorithm for tasks scheduling in multi-robot cells," in *Proc. 42nd Annu. Conf. IEEE Ind. Electron. Soc.*, 2016, pp. 692–697.

[27] F. Betti Sorbelli, S. Carpin, F. Coró, A. Navarra, and C. M. Pinotti, "Optimal routing schedules for robots operating in aisle-structures," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 4927–4933.

[28] A. Zelinsky *et al.* "Planning paths of complete coverage of an unstructured environment by a mobile robot," in *Proc. Int. Conf. Adv. Robot.*, vol. 13, pp. 533–538, 1993.

[29] I. A. Hameed, D. Bochtis, and C. A. Sϕrensen, "An optimized field coverage planning approach for navigation of agricultural robots in fields involving obstacle areas," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 5, 2013, Art. no. 231.

[30] T. Oksanen and A. Visala, "Coverage path planning algorithms for agricultural field machines," *J. Field Robot.*, vol. 26, no. 8, pp. 651–668, 2009.

[31] T. C. Thayer and S. Carpin, "Solving large-scale stochastic orienteering problems with aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2020, pp. 2452–2458.

[32] A. N. Shiryaev, *Optimal Stopping Rules*. Berlin, Germany: Springer Science & Business Media, 2007, vol. 8.

[33] T. H. Chung and J. W. Burdick, "Analysis of search decision making using probabilistic search strategies," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 132–144, Feb. 2012.

[34] F. B. Abdelaziz and S. Krichen, "Optimal stopping problems by two or more decision makers: A survey," *Comput. Manage. Sci.*, vol. 4, no. 2, pp. 89–111, 2007.

[35] G. Peskir and A. Shiryaev, *Optimal Stopping and Free-Boundary Problems*. Berlin, Germany: Springer, 2006.

[36] K. Chen and S. M. Ross, "An adaptive stochastic knapsack problem,"*Eur. J. OPR. Res.*, vol. 239, no. 3, pp. 625–635, 2014.

[37] G. Best, W. Martens, and R. Fitch, "Path planning with spatiotemporal optimal stopping for stochastic mission monitoring,"*IEEE Trans. on Robot.*, vol. 33, no. 3, pp. 629–646, Jun. 2017.

[38] J. Das *et al.*, "Data-driven robotic sampling for marine ecosystem monitoring," *Int. J. Robot. Res.*, vol. 34, no. 12, pp. 1435–1452, 2015.