Non-Monotone Energy-Aware Information Gathering for Heterogeneous Robot Teams

Xiaoyi Cai\textsuperscript{1,*}, Brent Schlotfeldt\textsuperscript{2,*}, Kasra Khosoussi\textsuperscript{1}, Nikolay Atanasov\textsuperscript{3}, George J. Pappas\textsuperscript{2}, and Jonathan P. How\textsuperscript{1}

Abstract—This paper considers the problem of planning trajectories for a team of sensor-equipped robots to reduce uncertainty about a dynamical process. Optimizing the trade-off between information gain and energy cost (e.g., control effort, energy expenditure, distance travelled) is desirable but leads to a non-monotonic objective function in the set of robot trajectories. Therefore, common multi-robot planning algorithms based on techniques such as coordinate descent lose their performance guarantees. Methods based on local search provide performance guarantees for optimizing a non-monotone submodular function, but require access to all robots’ trajectories, making it not suitable for distributed execution. This work proposes a distributed planning approach based on local search, and shows how to reduce its computation and communication requirements without sacrificing algorithm performance. We demonstrate the efficacy of our proposed method by coordinating robot teams composed of both ground and aerial vehicles with different sensing and control profiles, and evaluate the algorithm’s performance in two target tracking scenarios. Our results show up to 60% communication reduction and 80-92% computation reduction on average when coordinating up to 10 robots, while outperforming the coordinate descent based algorithm in achieving a desirable trade-off between sensing and energy expenditure.

I. INTRODUCTION

Developments in sensing and mobility have enabled effective utilization of robot systems in autonomous mapping \cite{1,2,3}, search and rescue \cite{4,5,6}, and environmental monitoring \cite{7,8,9}. These tasks require spatiotemporal information collection which can be achieved more efficiently and accurately by larger robot teams, rather than relying on individual robots. Robot teams may take advantage of heterogeneous capabilities, require less storage and computation per robot, and may achieve better environment coverage in shorter time \cite{10,11}. Task-level performance is usually quantified by a measure of information gain, where typically the marginal improvements diminish given additional measurements (submodularity), and adding new measurements does not reduce the objective (monotonicity). Although planning optimally for multi-robot sensing trajectories is generally intractable, these two properties allow for near-optimal approximation algorithms that scale to large robot teams, while providing worst-case guarantees. Additionally, practical implementations often need to consider various measures for energy expenditure, such as control effort, fuel cost, or distance travelled. A common approach is to impose fixed budgets, which preserves submodularity and monotonicity of the objective, so that existing algorithms may still be used \cite{12,13,14}.

In this paper, we are motivated by scenarios where robots, with potentially different sensing and control capabilities, have sufficient energy to spend but seek a desired trade-off between information gain and energy cost. Specifically, we formulate an energy-aware active information acquisition problem, where the goal is to plan trajectories for a team of heterogeneous robots to maximize a weighted sum of information gain and energy cost. One key observation with this approach is that adding the energy cost breaks the monotonicity of the objective, violating an assumption held by existing approximation algorithms. Therefore, we propose a new distributed planning algorithm based on local search \cite{15} (see Fig. 1) to achieve a worst-case performance guarantee for the non-monotone objective. We further develop techniques to reduce its computation and communication requirements compared to its naive distributed implementation, in order to scale to larger robot teams without affecting its theoretical guarantees.

Related Work. Our work belongs to the category of multi-robot...
informative path planning, where robots plan sensing trajectories to reduce uncertainty about a dynamic process, e.g., [2], [4], [16], [18], [20]–[23]. To alleviate the computational complexity which is exponential in the number of robots, various approximation methods have been developed to produce near-optimal solutions for a submodular and monotone objective (e.g., mutual information). A widely used technique is coordinate descent, or sequential allocation, where robots plan successively while incorporating the plans of previous robots [2], [16], [20], [21]. [16] showed that coordinate descent extends the near-optimality of a single-robot planner to the multi-robot scenario. Later, [24] extended the result to dynamics targets, showing that coordinate descent solution attains at least 50% of the optimal performance regardless of the planning order. With a more sophisticated order, [18] used coordinate descent to decentralize the greedy method [25], by only committing the best single-robot trajectory to the team solution in each sequential round. [4] proposed a distributed algorithm, where robots first independently generate plans, and choose subsets of those plans in a sequential fashion to greedily optimize the objective for a fixed number of rounds.

**Contributions.** The main limitation of the prior works is the assumption of monotonicity of the objective function. Problems without monotonicity, such as the energy-aware problem we propose, cannot be solved by the above methods while retaining their near-optimality properties. In contrast, our proposed algorithm provides a theoretical performance guarantee even for non-monotone objectives. In this work:

- We propose a distributed algorithm based on local search where robots collaboratively build a team plan by proposing modifications to the collective trajectories;
- We reduce its computation and communication requirements without affecting the performance guarantee;
- We demonstrate that the proposed algorithm outperforms a state-of-the-art algorithm for multi-robot target tracking in coordinating a team of heterogeneous robots, while trading off sensing performance and energy expenditure.

**II. Preliminaries**

We briefly review some useful definitions. Let \( g : 2^M \rightarrow \mathbb{R} \) be a set function defined on the ground set \( M \) consisting of finite elements. Let \( g(a|S) := g(S \cup \{a\}) - g(S) \) be the discrete derivative, or the marginal gain, of \( g \) at \( S \) with respect to \( a \).

**Definition 1 (Submodularity).** The function \( g \) is submodular if for any \( S_1 \subseteq S_2 \subseteq M \) and \( a \in M \setminus S_2 \), \( g(a|S_1) \geq g(a|S_2) \).

**Definition 2 (Monotonicity).** The function \( g \) is monotone if for any \( S_1 \subseteq S_2 \subseteq M \), \( g(S_1) \leq g(S_2) \).

**III. Problem Formulation**

Consider robots indexed by \( i \in \mathcal{R} := \{1, \ldots, n\} \) with states \( x_{i,t} \in \mathcal{X}_i \) at time \( t = 0, \ldots, T \). The robot dynamics are:

\[
x_{i,t+1} = f_i(x_{i,t}, u_{i,t}),
\]

where \( u_{i,t} \in \mathcal{U}_i \) is the control input and \( \mathcal{U}_i \) is a finite set. We denote a control sequence as \( \sigma_i = u_{i,0}, \ldots, u_{i,T-1} \in \mathcal{U}_i^T \).

The robots’ goal is to track targets with state \( y \in \mathbb{R}^d_y \) that have the following linear-Gaussian motion model:

\[
y_{t+1} = A_t y_t + w_t, \quad w_t \sim \mathcal{N}(0, W_t),
\]

where \( A_t \in \mathbb{R}^{d_y \times d_y} \) and \( w_t \) is a zero-mean Gaussian noise with covariance \( W_t \geq 0 \). The robots are equipped with sensors that measure the target state subject to an observation model:

\[
z_{i,t} = H_{i,t}(x_{i,t})y_t + v_{i,t}(x_{i,t}), \quad v_{i,t} \sim \mathcal{N}(0, V_{i,t}(x_{i,t})),
\]

where \( z_{i,t} \in \mathbb{R}^{d_z} \) is the measurement taken by robot \( i \) in state \( x_{i,t} \), \( H_{i,t}(x_{i,t}) \in \mathbb{R}^{d_z \times d_y} \), and \( v_{i,t}(x_{i,t}) \) is a state-dependent Gaussian noise, whose values are independent at any pair of times and across sensors. The observation model is linear in target states but can be nonlinear in the robot states. If it depends nonlinearly on the target state, we can linearize it around an estimate of the target state to get a linear model.

We assume every robot \( i \) has access to \( N_i \) control trajectories \( \mathcal{M}_i = \{\sigma_i^k\}_{k=1}^{N_i} \) to choose from. Denote the set of all control trajectories as \( \mathcal{M} = \bigcup_{i=1}^{N} \mathcal{M}_i \) and its size as \( N = |\mathcal{M}| \). Potential control trajectories can be generated by various single-robot information gathering algorithms such as [23], [26]–[28]. The fact that every robot cannot execute more than one trajectory can be encoded as a partition matroid \( (\mathcal{M}, \mathcal{I}) \), where \( \mathcal{M} \) is the ground set, and \( \mathcal{I} = \{S \subseteq \mathcal{M} \mid |S \cap \mathcal{M}_i| \leq 1 \forall i \in \mathcal{R} \} \) consists of all admissible subsets of trajectories. Given \( S \in \mathcal{I} \), we denote the joint state of robots that have been assigned trajectories as \( x_{S,t} \) at time \( t \), and their indices as \( \mathcal{R}_S := \{i \mid |\mathcal{M}_i \cap S| = 1 \forall i \in \mathcal{R} \} \). Also, denote the measurements up to time \( t \leq T \) collected by robots \( i \in \mathcal{R}_S \) who follow the trajectories in \( S \) by \( z_{S,t} \).

Due to the linear-Gaussian assumptions in (2) and (3), the optimal estimator for the target states is a Kalman filter. The target estimate covariance \( \Sigma_{S,t} \) at time \( t \) resulting from robots \( \mathcal{R}_S \) following trajectories in \( S \) obeys:

\[
\Sigma_{S,t+1} = \rho^p_S(\Sigma_{S,t}, x_{S,t+1}),
\]

where \( \rho^p(\cdot) \) and \( \rho^p_S(\cdot, \cdot) \) are the Kalman filter prediction and measurement update, respectively:

**Predict:**

\[
\rho^p_S(\Sigma) := A_t \Sigma A_t^\top + W_t,
\]

**Update:**

\[
\rho^p_S(\Sigma, x_{S,t}) := \left( \Sigma^{-1} + \sum_{i \in \mathcal{R}_S} M_{i,t}(x_{i,t}) \right)^{-1},
\]

\[
M_{i,t}(x_{i,t}) := H_{i,t}(x_{i,t})V_{i,t}(x_{i,t})^{-1} H_{i,t}(x_{i,t})^\top.
\]

The robots are assumed to know each other’s dynamics (1) and observation models (3) so that they can assess the overall tracking performance given others’ trajectories.

When choosing sensing trajectories, we want to capture the trade-off between sensing performance and energy expenditure, which is formalized below.

**Problem 1 (Energy-Aware Active Information Acquisition).** Given initial states \( x_{i,0} \in \mathcal{X}_i \) for every robot \( i \in \mathcal{R} \), a prior distribution of target state \( y_0 \), and a finite planning horizon \( T \), find a set of trajectories \( S \in \mathcal{M} \) to optimize the following:

\[
\max_{S \in \mathcal{I}} J(S) := \mathbb{I}(y_{1:T}; z_{S_{1:T}}) - C(S),
\]
where \( I(y_{1:T}; z_{S_1:T}) = \frac{1}{2} \sum_{t=1}^{T} \left[ \log \det \left( \rho_{t-1}(\Sigma_{S_t,t-1}) \right) - \log \det(\Sigma_{S_t,t}) \right] \geq 0 \) is the mutual information between target states and observations, and the energy objective \( C : 2^\mathcal{M} \to \mathbb{R} \) is defined as:

\[
C(S) := \sum_{\sigma_i \in S} r_i C_i(\sigma_i),
\]

where \( r_i \geq 0 \) is a tuning parameter, and \( 0 \leq C_i(\cdot) \leq C_{i,\text{max}} \) is a non-negative, bounded energy cost for robot \( i \) to apply control sequence \( \sigma_i \).

**Remark 1.** The optimization problem (5) is non-monotone, because adding extra trajectories may worsen the objective by incurring high energy cost \( C(S) \). Thus, the constraint \( S \in \mathcal{I} \) may not be tight, i.e., some robots may not get assigned trajectories. This property is useful when a large repository of heterogeneous robots is available but only a subset is necessary for the given tasks.

**Remark 2.** The choice of (5) is motivated by the energy-aware target tracking application. However, the proposed algorithm in Sec. IV is applicable to any scenario where \( J(S) \) is a submodular set function that is not necessarily monotone, but can be made non-negative with proper offset.

Solving Problem 1 is challenging because adding energy cost \( C(S) \) breaks the monotonicity of the objective, a property required for approximation methods such as coordinate descent [2] and the greedy algorithm [25] to maintain their performance guarantees. This is because these methods only add elements to the solution set, which always improves a monotone objective, but can worsen the objective in our setting, and may yield arbitrarily poor performance. We now propose a new distributed algorithm for Problem 1 based on local search [19].

### IV. Multi-Robot Planning

We first present how local search [19] can be used to solve Problem 1 with near-optimal performance guarantee. Despite the guarantee, local search is not suitable for distributed robot teams, because it assumes access to all locally planned robot control trajectories which can be communication-expensive to gather. To address this problem, we propose a new distributed algorithm that exploits the structure of a partition matroid to allow robots to collaboratively build a team plan by repeatedly proposing modifications to the collective trajectories. Moreover, we develop techniques to reduce computation and communication without affecting the performance guarantees.

In the following subsections, we denote \( g : 2^\mathcal{M} \to \mathbb{R} \) as the non-negative, submodular oracle function used by local search, where the ground set \( \mathcal{M} \) consists of the robot trajectories.

#### A. Centralized Local Search (CLS)

We present the original local search [19] in our setting with a single partition matroid constraint. We refer to it as centralized local search (CLS) (Alg. 1) because it requires access to trajectories \( \mathcal{M} \) from all robots. The algorithm proceeds in two rounds to find two candidate solutions \( S_1, S_2 \in \mathcal{I} \). In each round \( k = 1, 2 \), solution \( S_k \) is initialized with a single-robot trajectory maximizing the objective (Line 5). Repeatedly, \( S_k \) is modified by executing one of the Delete, Add or Swap operations, if it improves the objective by at least \( (1 + \frac{\alpha}{N'}) \) of its original value (Lines 6-9), where \( \alpha > 0 \) controls run-time and performance guarantee. This procedure continues until \( S_k \) is no longer updated, and the next round begins without considering \( S_k \) in the ground set \( \mathcal{M} \) (Line 10). Lastly, the one between \( S_1 \) and \( S_2 \) that results in better objective is returned.

**Proposition 1.** Consider that we solve Problem 1 whose objective is made non-negative by adding a constant offset:

\[
\max_{S \in \mathcal{I}} \ g(S) := J(S) + O,
\]

where \( O := \sum_{i=1}^{n} r_i C_{i,\text{max}} \). Denote \( S^* \) and \( S^{ls} \) as the optimal solution and solution obtained by CLS (Alg. 1) for (7), by using \( g(\cdot) \) as the oracle. We have the following worst-case performance guarantee for the objective:

\[
0 \leq g(S^*) \leq 4(1 + \alpha)g(S^{ls}).
\]

**Proof.** In (5), mutual information is a submodular set function defined on measurements provided by selected trajectories [2]. Furthermore, \( C(S) \) is modular due to its additive nature:

\[
C(S) = \sum_{\sigma_i \in S} r_i C_i(\sigma_i) \geq 0.
\]

Since mutual information is non-negative, (7) is a submodular non-monotone maximization problem with a partition matroid constraint. The proposition follows from [19, Thm. 4].

**Remark 3.** Having the constant \( O \) term in (7) does not change the optimization in Problem 1, but ensures that the oracle used by CLS (Alg. 1) is non-negative so that the ratio \( (1 + \frac{\alpha}{N'}) \) correctly reflects the sufficient improvement condition.

Besides the communication aspect that CLS requires access to all robot trajectories, running it naively can incur significant
B. Distributed Local Search (DLS)

In this section, we propose a distributed implementation of the local search algorithm to address both the communication and computation concerns. The algorithm is presented in Alg. 2 and Alg. 3 written from robot $i$’s perspective. Exploiting the structure of the partition matroid, DLS allows every robot to propose local operations based on its own trajectory set, while guaranteeing that the team solution never contains more than one trajectory for every robot. In fact, all steps executed by CLS can be distributedly proposed by robots, thus providing the same performance guarantee shown in Theorem 1.

1) Distributed Proposal: Every proposal consists of two trajectories $(d, a)$, where $d$ is to be deleted from and $a$ is to be added to the solution set. We also define a special symbol “NOP” that leads to no set operation, i.e., $S_k \cup \{\text{NOP}\} = S_k \setminus \{d\} = S_k$. Note that $(d, \text{NOP})$, $(\text{NOP}, a)$ and $(d, a)$ are equivalent to the Delete, Add and Swap steps in CLS.

Every robot $i$ starts by sharing size of its trajectory set $|M_i|$ and its best trajectory $a^* \in M_i$ in order to initialize $N$ and $S_k$ collaboratively (Alg. 2 Lines 5-7). Repeatedly, every robot $i$ executes the subroutine FindProposal (Alg. 3) in parallel, in order to propose changes to $S_k$. Since any valid proposal shared by robots will improve the objective, the first $(d, a) \neq \text{(NOP, NOP)}$ will be adopted by every robot to update $S_k$ in every round (Alg. 2 Lines 8-10). We assume instantaneous communication, so robots always use a common proposal to update their local copies of $S_k$. Otherwise, if delay leads to multiple valid proposals, a resolution scheme is required to ensure robots pick the same proposal.

In FindProposal (Alg. 3), an outer loop looks for potential deletion $d \in S_k$ (Alg. 3 Lines 2-6). Otherwise, further adding $a \in M_i$ is considered, as long as the partition matroid constraint is not violated (Alg. 3 Lines 7-8). Next, we discuss how to efficiently search for trajectories to add.

2) Lazy Search: Instead of searching over trajectories in an arbitrary order, we can prioritize the ones that already perform well by themselves, based on $g(a|\emptyset)$ for all $a \in M_i$ (Alg. 2 Line 2). In this fashion, we are more likely to find trajectories that provide sufficient improvement earlier (Alg. 3 Lines 12-13). Note that $g(a|\emptyset)$ is typically a byproduct of the trajectory generation process, so it can be saved and reused.

This ordering also allows us to prune unpromising trajectories. Given the team solution after deletion $S_k^- := S_k \setminus \{d\}$, the required marginal gain for subsequently adding trajectory $a$ is

$$g(a|S_k^-) \geq \Delta := (1 + \frac{\alpha}{N^4}) g(S_k) - g(S_k^-).$$

We can prune any $a \in M_i$, if $g(a|\emptyset) < \Delta$ based on the diminishing return property: because $\emptyset \subseteq S_k^-$, we know that

$$\Delta > g(a|\emptyset) \geq g(a|S_k^-),$$

violating condition (10). Similarly, all subsequent trajectories $a'$ can be ignored, because their marginal gains $g(a'|\emptyset) \leq g(a|\emptyset) < \Delta$ due to ordering (Alg. 3 Lines 10-11). Lastly, if adding a trajectory improves $S_k^-$ sufficiently, the proposal is broadcasted (Alg. 3 Lines 12-13).

3) Greedy Warm Start: We observe empirically that a robot tends to swap its own trajectories consecutively for small growth in the objective, increasing communication unnecessarily. This can be mitigated by a simple technique: when finding local operations initially, we force robots to only propose additions to greedily maximize the objective, until doing so does not lead to enough improvement or violates the matroid constraint. Then robots resume normally with Alg. 3, allowing all local operations. By warm starting the team solution greedily, every robot aggregates numerous proposals with smaller increase in the objective into a greedy addition with larger increase, thus effectively reducing communication.

V. SIMULATION RESULTS

We evaluate DLS in two target tracking scenarios based on objective values, computation, communication, and ability to handle heterogeneous robots. Its performance is compared against coordinate descent (CD) [2], a state-of-the-art algorithm for multi-robot target tracking that, however, assumes monotonicity of the objective. Planning for robots sequentially, CD allows every robot to incorporate the plans of previous robots. We also allow CD to not assign any trajectory to a robot.
The quadrotors are restricted to a plane to avoid collisions. Comparisons between CLS and DLS are omitted because the two algorithms empirically achieve the same average performance. We set $\alpha = 1$ arbitrarily, because tuning it was not effective due to the large number of trajectories $N$.

Both DLS and CD are implemented in C++ and evaluated in simulation on a laptop with an Intel Core i7 CPU. For DLS, every robot owns separate threads, and executes Alg. 3 over 4 extra threads to exploit its parallel structure. Similarly, CD allows every robot to use 4 threads and additionally incorporates accelerated greedy [29] for extra speed-up.

A. Robots Characteristics

Given initial state $x_{i,0} \in A'_i$ for robot $i \in \mathcal{R}_S$ who follows the control sequence $u_{i,0}, \ldots, u_{i,T-1} = \sigma_i \in S$, the resultant states are $x_{i,1}, \ldots, x_{i,T}$ based on dynamics (1). The energy cost $C(S)$ may also be state-dependent. We define it as:

$$C(S) := \sum_{i \in \mathcal{R}_S} \sum_{t=0}^{T-1} r_i \left( c_{i,1}^{ctrl}(u_{i,t}) + c_{i,t}^{state}(x_{i,t}) \right),$$

(11)

where the state-dependent cost $c_{i,t}^{state}()$ and control-dependent cost $c_{i,1}^{ctrl}()$ are defined based on robot types—in our case, robot $i$ is either an unmanned ground vehicle (UGV) or an unmanned aerial vehicle (UAV). Note that decomposition between state and control is not required for our framework to work. The setup for robots are summarized in Table I. For simplicity, all robots follow differential-drive dynamics with sampling period $\tau = 0.5$ and motion primitives consisting of linear and angular velocities $\{ u = (\nu, \omega) \mid \nu \in \{0, 8\} \text{ m/s, } \omega \in \{0, \pm \frac{\pi}{6}\} \text{ rad/s} \}$. We consider muddy and windy regions that incur state-dependent costs for UGVs and UAVs respectively. The robots are equipped with range and bearing sensors, whose measurement noise covariances grow linearly with target distance. The sensors have limited ranges and field of views (FOVs), within which the maximum noise standard deviations are 0.1 m and $5^\circ$ for range and bearing measurements, respectively. Outside the range or field of view, measurement noise becomes infinite. Refer to [20] for details.

B. Scenario 1: Multi-Robot Dynamic Target Tracking

The performance, computation and communication requirements of DLS are evaluated and compared against CD (see Fig. 2 and 3). The scenario involves 2, ..., 10 UGVs trying to estimate the positions and velocities of the same number of dynamic targets that follow discretized double integrator dynamics. Targets have a top speed of 2 m/s and their models are corrupted by Gaussian noise. Robots and targets are spawned in a square arena whose sides grow from 40 m to 60 m, and 50 random trials are run for each number of robots.

Non-monotonicity in the problem is accentuated by an increasing penalty for control effort of additional robots, by setting $r_i = \bar{i}$ for each robot $i$ as defined in (11) (i.e., the 10-th added robot is 10 times more expensive to move than the first). Note that state-dependent cost is set to 0 only for this experiment. Trajectory generation has parameters $\epsilon = 1$ and $\delta = 2$ for horizon $T = 10$. As the planning order is arbitrary for CD, we investigate two planning orders: first from cheaper to more expensive robots, and the reverse. Intuitively and shown in Fig. 3, the former should perform better than the later, because the same amount of information can be gathered

### Table I

| Robot Setup in Two Experimental Scenarios | $c^{ctrl}(u)$, $u$ given as | $c^{state}(x)$, $x$ in | FOV ($^\circ$) | Range (m) |
|-----------------------------------------|-----------------------------|------------------------|---------------|----------|
|                                         | 0, 0, $\pm\frac{\pi}{8}$  | 8, $\pm\frac{\pi}{4}$ | Exp.1&2       | Exp.1&2   |
| UGV                                     | 0                           | 1                      | 2             | 160      | 6 & 15   |
|                                         |                             | 3                      | /             |          |
| UAV                                     | 2                           | 2                      | 4             | 360      | / & 20   |
|                                         |                             | /                      | 3             |          |
while spending less energy. While other orderings are possible (e.g., \cite{18} uses CD to decentralize greedy algorithm), we only use two to show CD’s susceptibility to poor planning order.

Proposed methods for improving naive distributed execution of local search, namely lazy search (Lazy) and greedy warm start (Warm), are shown to reduce computation by 80-92\% and communication by up to 60\% on average, as shown in Fig. 2. As expected, when there are few robots with similar control penalties, the objective is still close to being monotone and DLS and CD achieve similar performance as shown in Fig. 3. However, as more costly robots are added, their contribution in information gain is reduced by high control penalty, the problem becomes more non-monotone. Therefore, the performance gap between DLS and CD widens, because CD requires monotonicity to maintain its performance guarantee, but DLS does not. From Fig. 3, we can see that planning order is critical for CD to perform well, but a good ordering is often unknown a priori. Compared to CD which requires only \(n - 1\) communication rounds for \(n\) robots, DLS requires more for its performance. For practical concerns to save more time, DLS with down-sampled trajectories (e.g., keeping the best 10\% of each robot’s trajectories) still produces better solution than CD, but the guarantee of DLS no longer holds.

\section*{C. Scenario 2: Heterogeneous Sensing and Control}

Here we consider a heterogeneous team consisting of 2 UGVs and 1 UAV with different sensing and control profiles (Table I) in order to track 10 static targets in a 100 m \(\times\) 100 m arena, over a longer horizon \(T = 20\). The UAV has better sensing range and field of view compared to UGVs, but consumes more energy. The arena has overlapping muddy and windy regions, so robots have to collaboratively decide who should venture into the costly regions. To explore trade-off between sensing and energy objectives as a team, we set a common \(r_i = r\) for every robot \(i\). We vary \(r\) from 0 to 0.5 and run 50 trials for each fixed value. Robots are spawned in the non-muddy, non-windy region, but targets may appear anywhere. Different from last scenario, we set \(\delta = 4\) to handle the longer horizon. We simply let CD plan for the UAV first, since it is unclear what a good ordering is when robots do not dominate each other in both energy cost and sensing ability.

As shown in Fig. 4, DLS consistently achieves better sensing and energy trade-off than CD on average. To gain intuitions on why CD under-performs, a particular trial given \(r = 0.2\) is shown in Fig. 5. Due to the non-monotone objective, the robot who plans first to maximize its own objective can hinder robots who plan later, thus negatively affecting team performance.

\section*{VI. CONCLUSION}

This work considered a multi-robot information gathering problem with non-monotone objective that captures the trade-off between sensing benefits and energy expenditure. We proposed a distributed algorithm based on local search and reduced its computation and communication requirements without affecting its performance guarantees. The proposed algorithm was evaluated in two target tracking scenarios and outperformed the state-of-the-art coordinate descent method. Future work will focus on scaling the algorithm to large robot teams by exploiting spatial separation, formalizing heterogeneity, and carrying out hardware experiments.

\section*{ACKNOWLEDGMENT}

This research was supported in part by Boeing Research & Technology and ARL DCIST CRA W911NF-17-2-0181. The authors would like to thank ACL members for their valuable discussions and feedback.
