Multi-Agent Intermittent Interaction Planning via Sequential Greedy Selections Over Position Samples

Larkin Heintzman, and Ryan K. Williams

***Abstract*—In this work, we propose a method to solve the *interaction planning* problem for a set of mobile agents with obstacles and agent collisions via a core path planner and constrained random position sampling approach. The interaction constraint is posed in the form of an arbitrary number of discretized times in which we enforce a desired topological condition. The general objective function, to be maximized subject to the interaction constraint, is coverage of an environmental process here modeled as a Gaussian mixture model. The main tool we use to select positions and the times of interaction is the greedy algorithm, along with a submodular objective function and matroid constraint. Through this we guarantee strong theoretical lower bounds on sub-optimality. Simulations, including several Monte Carlo trials, are presented to corroborate our proposed methods.**

***Index Terms*—Surveillance Robotic Systems, Path Planning for Multiple Mobile Robots or Agents, Multi-Robot Systems**

I. Introduction

MULTI-agent systems (MASs) are of great interest to researchers because they allow difficult problems to be broken down into more manageable parts. One such problem that we consider in this paper is environmental monitoring. In such a problem, where a single robot may fail, a *team* of multiple robots may excel, often by exploiting *interaction*. The question then becomes, how does one guide a MAS to accomplish its objectives, such as monitoring an environment, subject to some *constraints* on agent-to-agent interaction?

This question can take many forms for various applications above ground, underwater, or in static sensor networks. In [1], [2], this question takes the form of maintaining *observability* of the agents' states during underwater mapping [3]. While in [4] the tasks of localization and objective function improvement are split up over the agent set, reinforcing the idea that MASs are adaptable to many different scenarios. MASs can also be found in the field of wireless power transfer (WPT) [5]–[7], where the goal is to generate a network of wireless charging stations subject to the constraint of graph connectivity.

While multi-agent interaction has been investigated quite deeply, *planning* with interaction constraints poses a challenge due to general intractability and limited computational resources in fielded systems [8]. To cope with intractable planning problems, researchers have developed methods that maintain tractability, but also give performance guarantees compared to the optimal solution [5]–[7], [9], [10]. An example of a scalable method in a combinatorial optimization setting is the *greedy* algorithm which, when paired with a submodular set function and a matroid constraint, generates strong approximation ratios [10]–[12]. While other, more computationally complex algorithms have been developed for multi-agent planning with interaction constraints [13], in this work we use the greedy approach, tailored to our setting of environmental monitoring with interaction constraints.

Specifically, we propose a method to solve the *interaction planning* problem for a set of mobile agents, with predetermined starting and goal locations, via a constrained random position sampling approach while avoiding obstacles in the environment. Agents have fixed sensing and interaction radii, giving rise to interaction links between agents, known as a *proximity graph*. We discretize time into a set of *time slices*, and pose the interaction constraint in the form of an arbitrary number of time slices to achieve a desired topological condition. We choose completeness of the interaction graph as it allows for one-hop communication between agents. The objective function, to be maximized under given constraints, is coverage of an environmental process modeled as a Gaussian mixture model (GMM). Here we are interested in coverage via *direct* paths to the goal locations, rather than potentially meandering paths to maximize coverage [14]. Maximizing coverage while maintaining intermittent communication is challenging because we are forced to reduce coverage to meet the constraint.

The motivation for our problem comes from practical considerations of battery life, computation resources, and data storage on-board mobile robots in a search and rescue scenario. During a monitoring application, agents rapidly deplete their reserves of battery life and data storage space. Rather than ceasing the operation to offload data, in such cases the agents could re-distribute the information among themselves if the opportunity were available. Also predefined entry and exit points allow human operators to better utilize the autonomous system as a whole. In this context, we address the following question: *how do we efficiently determine when and where it is appropriate for a multi-agent team to interact, in response to team objectives and the environment?*

Here we discuss works that have solved problems related to the current one, while noting our novelties. The problem of *intermittent connectivity* has been approached via linear temporal logic (LTL) [15] very recently, and the method has

Manuscript received: Aug, 20, 2020; Revised Nov, 17, 2020; Accepted Dec, 17, 2020.

This paper was recommended for publication by Editor M. Ani Hsieh upon evaluation of the Associate Editor and Reviewers' comments. *This work was supported by the National Science Foundation through grant # CNS-1830414

L. Heintzman and R. K. Williams are with the Department of Electrical and computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA USA {hlarkin3, rywilli1}@vt.edu

Digital Object Identifier (DOI): see top of this page.

yielded powerful results, however with significant complexity. The main result of [15] is that of robot-to-robot information exchange occurring infinitely often while accomplishing independent tasks. However, in [15] during initialization each agent must solve a potentially large number of *control synthesis* problems. The number of problems for a single agent grows with the set of meeting locations available to each team the agent belongs to, which could be quite large. The control synthesis problems can be derived using graph search techniques, so a single synthesis can be computationally intensive. Instead, our method exploits a greedy algorithm, along with sampling-based path planning methods, which allows us to make a trade off of less representational power for vastly improved computational scaling [10], [16]. In addition our method explicitly deals with, and takes advantage of, coverage-based task execution rather than the general tasks treated in [15].

In [13], and in the expanded work [17], the authors consider multi-robot informative path planning (MIPP) with *periodic connectivity*, and derive an algorithm with strong performance guarantees. However, the method proposed in [13] is computationally complex, being a form of coordinate ascent in the robot path space, requiring multiple iterations of path planning to converge and a discretized environment graph. The running time of the algorithm in [13] is *exponential* in the depth of the planned paths, meaning that longer periods between re-planning results in large computation times. Both our method and [13] consider obstacles, however we explicitly deal with inter-agent collisions and goal locations. In addition, our method does not require multiple iterations to converge nor does it scale exponentially with path length/depth. The algorithm we propose relies on multiple greedy selections, unlike any previous work in this area [13], [15], [17], [18]. Later we will present simulations comparing the proposed method to the method of [13] and it's extension [17].

Finally, a related problem known as the *multirobot connectivity maintenance* problem is first addressed in [19], and later in [18], [20]. In these works, the goal is to optimally redeploy a team of robots from starting locations, in a graph-represented environment, such that they form a connected communication subgraph while minimizing distance traveled. In [20], connected goal locations for robots are selected via ILP[1] based on the environment and a distance-based objective function. In our problem, however, intermediate locations are instead determined by the change in environmental process coverage that results from the interaction, and we also focus on approximate solutions with guarantees rather than optimal ones. It is also worth noting that we plan multi-agent paths while using a more stringent form of connectivity, one-hop communication or completeness, as compared to other works in this area [13], [15], [18], [20]. Using a *complete* graph does constrain the planning space compared to the general problem, which has a PSPACE-complete complexity result for multi-agent *connected* path planning (MCPP) [13], [17], [21]. However, in our complete graph approach we retain complex temporal constraints as well as suboptimality guarantees.

[1]Integer linear programming (ILP), where instances were solved with a commercial solver GUROBI.

The remainder of this work is organized as follows, in Section II we discuss background information and formally introduce the problem. In Section III we discuss the position sampling process, and the greedy algorithm. In Section IV we discuss selecting the times of interaction, and derive a lower bound on performance. In Section V we present in depth simulations to demonstrate the effectiveness and adaptability of the results derived, as well as comparisons to a similar solution. Lastly, in Section VI we conclude this work.

II. Preliminaries

We will model the interaction planning problem as a *time of interaction* (TOI) selection problem. As such we define a set of discretized times from which to make selections, denoting the set of time indexes as $T_t = \{1, 2, \dots, T_{end}\}$, where $k \in T_t$ is a single time index, and $T_{end} \in \mathbb{N}$ is the final time. In defining uniformly discretized times, it is expected that the coarseness of discretization will effect solution quality, however this is expected and a natural trade off. From T_t we select $K \leq |T_t|$ time slices, during which agents must form a complete interaction graph. We take Assumption 1 regarding selection from T_t. Assumption 1 will be of critical importance in deriving an efficient algorithm. However, the assumption also implies a certain understanding an agent's feasible paths, to be discussed in more detail in Section III.

Assumption 1. Time slices in T_t are considered independently from one another, meaning that interacting in an arbitrary time slice does not impact future choices. Formally, any $T \subseteq T_t$ with $|T| \geq K$ is considered a feasible solution to the problem, regardless of the times in T.

In this work, the interaction constraint will be *completeness* of the agent interaction graph, meaning that all agents can communicate with each other. We choose completeness as it implies one-hop communication between all agents, and thus maximizes information exchange [22]. The TOI selection method proposed in this work, to be discussed in Section IV, is a general model for planning multi-agent interactions. As such, other constraints could be used. For example, in the case of connectivity, several algorithms exist that have a lower bounded disc-based coverage performance [5]–[7] any of which could be used in this work. Note however, the value of the lower bound would affect the approximation ratio.

Next, let us define the sampled 2D positions from which a MAS trajectory is built. For every time $k \in T_t$, and every $i \in \mathcal{A}$, with $\mathcal{A} = \{1, 2, \dots, N\}$ being the set of agent indexes, let us define $\mathcal{V}_{ki} = \left\{ x_{ij}^{(k)} \in \mathbb{R}^2 \,|\, \forall j \in T_p \right\} \subset \mathbb{R}^{2\times|T_p|}$ as the set of all sampled positions *available* to agent i at time k, where we represent the set of position indexes with T_p, with $|T_p|$ samples available to *each* agent. A single position sample $x_{ij}^{(k)}$ indicates a 2D position, available to agent $i \in \mathcal{A}$ during trajectory planning, with position index $j \in T_p$, at time slice $k \in T_t$. Let us also define $\mathcal{V}_k = \{x \in \mathcal{V}_{ki} \,|\, \forall i \in \mathcal{A}\}$ as the positions available to *all* agents at time k. The specifics on how these samples are generated will be discussed in Section III. The team of agents is also given starting locations, as $\mathcal{V}_1$,

and goal locations, as $\mathcal{V}_{T_{end}}$, both of which are assumed to be given a priori. Both $\mathcal{V}_1$ and $\mathcal{V}_{T_{end}}$ can be thought of as sets, at the starting and ending times, with exactly N pre-selected positions each.

In the coming sections, we will derive a method for selecting $|T_t|$ *sub-goal positions* for each agent, one for each time slice, while respecting the completeness interaction constraint at K time slices and avoiding obstacle and inter-agent collisions. This sequence of positions can be thought of as a list of intermediate goal positions along some path between the team's starting locations and the team's goal locations. In Section V we show one possible method, from [23], for linking these samples to generate a path that takes into account collision and obstacle avoidance. Of course the method of [23] is not the only way to generate fully realized paths, but it is convenient for our purposes.

A. Agent Modeling

Here we discuss the agent model and sensing capabilities. The stacked system vector of all agents is $\mathbf{x}_k = \{p_{1k}^\top, p_{2k}^\top, \dots, p_{Nk}^\top\}^\top \in \mathbb{R}^{2N}$, where $p_{ik} \in \mathbb{R}^2$ is the *position* of agent i at time slice k. We assume a generic motion model for all agents $\dot{\mathbf{x}}_k = f(\mathbf{x}_k)$ through which the agents' motion can be controlled. In addition, each agent has an *interaction radius*, ρ, and a *sensing radius*, δ, which are the maximum distance at which agents can communicate and gain information about the environment, respectively. Since we wish to generate dynamically feasible paths for all agents, we take the following assumption regarding agent motion.

Assumption 2. Agents using the motion model, as described by $\dot{\mathbf{x}}_k = f(\mathbf{x}_k)$, can execute dynamically feasible paths with an upper bounded tracking error, with the upper bound of tracking error being known a priori.

Assumption 2, inspired by [23], is not highly limiting because, given a known upper bound on tracking error, we can perform the proposed method within a configuration space (increasing the size of obstacles and agent collision radii accordingly) [9]. Further, the upper bound of tracking error can be chosen quite conservatively if required.

B. Interaction and Environmental Modeling

Agent interactions are modeled by a graph, as is common in MAS literature [22], [24]. We define graph $G_k = (\mathcal{V}_k, \mathcal{E}_k)$, with position samples as vertices, for every $k \in T_t$. Further, $e = (u, v) \in \mathcal{E}_k \iff \|u - v\| \leq \rho$, where $u, v \in \mathcal{V}_k$ are *position samples* at time k. Each graph G_k is undirected, symmetric, static, and without self-loops. During interaction we require that $S \subseteq \mathcal{V}_k$ induces a *complete* induced subgraph $G_k(S)$, where all edges exist.

The environment, $E \subset \mathbb{R}^2$, is a subset of the plane that contains all objects of interest, obstacles, and an environmental process, modeled as a GMM as in [25], that we would like to sense. The environment can represent a physical space of interest, or a configuration space depending upon the application. To define the GMM, a set of means are chosen, $\mu = \left[\mu_1^\top, \dots, \mu_m^\top\right]^\top$, where each $\mu_j = [\mu_{j_x}, \mu_{j_y}]^\top \in \mathbb{R}^2$, where m is the number of means. A set of covariances are also selected $\sigma^2 = [(\sigma_1^2)^\top, \dots, (\sigma_m^2)^\top]^\top$, where each $\sigma_j^2 = [\sigma_{j_x}^2, \sigma_{j_y}^2]^\top \in \mathbb{R}^2$. We assume the mixture weights sum to unity to correctly define the PDF.

We assume that the environment, obstacles, and GMM are known a priori. The methods we derive also apply to scenarios where the process model is being *learned*. For example, the GMM can be redefined to represent points of highest *uncertainty* in some unknown process [22]. In the case where we have zero a priori information, a uniform distribution can be used instead.

C. Multi-Agent Objective

To gain information about an environmental process, we define an objective function based on sensing radius, agent position, and the process itself. Let us define a function $z : 2^{\mathcal{V}_k} \to \mathbb{R}^2$, that maps samples to a subset of the plane:

$$z(S \subseteq \mathcal{V}_k) = \bigcup_{u \in S} N_E^\delta(u) \tag{1}$$

where $N_E^\delta(u)$ indicates the δ-radius closed neighborhood in E centered at the sample $u \in S$. Now we can define the objective function $g : 2^{\mathcal{V}_k} \to \mathbb{R}$ as:

$$g(S \subseteq \mathcal{V}_k) = f_E(z(S)) \tag{2}$$

where $f_E : \mathbb{R}^2 \to \mathbb{R}$ is the coverage of the GMM in some subset of the plane. Here $z(S)$ gives a subset of the plane, and $f_E(z(S))$ gives the GMM coverage associated with that subset. The coverage function is defined as:

$$f_E(P \subseteq \mathbb{R}^2) = \iint_P f_{pdf}(x, y) dA \tag{3}$$

where $f_{pdf}(x, y)$ is the PDF of the GMM, and dA represents integration over area P (i.e. $dA = dxdy$). The effect of (2) is that more benefit is gained the more probability mass is within the sensing radii of agents. A result of this objective is that the optimal positions will tend to not overlap in terms of *sensing* radius, creating a trade-off between the topological constraint and coverage. Set coverage is well known to be a submodular function [10], [26], which we will later exploit to derive an efficient interaction planner.

D. Problem Statement

We state the problem considered in this work, and discuss the impact of assumptions.

Problem 1. Given agent set $\mathcal{A}$ satisfying Assumption 2, known environment E, and position sample set $\mathcal{V}_k$, find N position samples for the agents by solving:

$$S = \underset{S \subseteq \mathcal{V}_k,\, |S| = N}{\arg\max}\ g(S) \tag{4}$$

Maximizing $g(S)$, from (2), by selecting a set S for each time $k \in T_t$ subject to the interaction constraint:

$$\begin{gathered} \sum_{k \in T_t} \mathbb{K}\Big(G_k(S)\Big) \geq K, \text{ where} \\ \mathbb{K}(G) = \begin{cases} 1, & \text{if } G \text{ is complete} \\ 0, & \text{otherwise} \end{cases} \end{gathered} \tag{5}$$

requiring agents to form a complete graph in at least K time slices, under Assumption 1.

The result of solving Problem 1 is a selection of position samples, each with an associated time. These samples maximize coverage of the environmental process subject to an intermittent interaction constraint.

We make several assumptions in solving this problem. The the path tracking error is assumed to be known, Assumption 2, allowing us to apply the current work to a much wider variety of application areas by specifying which robotic motion models could be used. The environment is assumed to be known fully, which includes knowledge of the obstacles as well as the environmental process. It is certainly possible for a higher dimension configuration space to be used, rather than a 2D space we use here. Although assuming a known environment is certainly significant, we argue that it does not trivialize the problem since [17] takes a similar assumption, and that extensions are straightforward.

Regarding the independence of time slices, Assumption 1, it allows us to model the TOI selection as a modular minimization problem [11], while retaining the representational power of time indexed MAS trajectories. An implication of Assumption 1 is trajectory feasibility being guaranteed. The implication can be restated as the next sample being reachable from any point in the previous bounding set, details in Section III-A. In practice, this feasibility requirement will be satisfied via a *core* path that guarantees each agent's path is feasible without collisions.

III. Generating Multi-Agent Position Samples

Position samples for time $k \in T_t$ are generated within a *bounding set*, B_k, of the plane via a uniform distribution and feasibility check. We discuss the method of constructing these bounding sets for two cases in each time slice, unconstrained and constrained, and how they are placed with regard to a *core* path. Once generated, we impose a matroid constraint on the selection of samples then detail the greedy algorithm as it pertains to sample selection.

A. Bounding Set Selection

Here we discuss the bounding sets that guide the position sampling process, and determine the extents of agent paths. A bounding set, B_k for time slice k, allows us to generate samples within it and to control agent interaction. In order to ensure interaction we require conditions on each bounding set to force the desired constraint. However, at an arbitrary time $k \in T_t$ it is unknown during sample generation whether agents are interacting. Thus we define two bounding sets for each k, one that allows coverage B_k, and one to ensure the interaction constraint $\overline{B}_k$. We give explicit definitions for each case.

1) Unconstrained Bounding Set B_k: We begin by generating a so-called core path from the team's mean starting location to the team's mean goal location, creating a reference for individual agents to plan around. We employ CL-RRT*, seen in [9], [23], to plan the core path (we use the same method later to plan agent paths). We label the core path $C_p \in \mathbb{R}^{2\times L}$ where L is the number of points along the path. Further, we take the mild assumption that each agent has a feasible path from their starting location to the *beginning* of C_p, similarly for their *end* of C_p and goal location. This assumption is not highly limiting because it is easily satisfied by having no obstacles directly separating the agent's starting locations, and similarly for the agent's goal locations. See Figure 1 for an example of a core path, and starting/goal locations satisfying the previous assumption.

The next step is to define bounding set $B_k \subseteq E$, a rectangular area of width r and height h, centered on the 2D point c_k, for each time slice k. The rectangle is oriented parallel to C_p at each point c_k, such that the width r is in-line with C_p. Each center point c_k is selected at distance intervals of $\frac{\|C_p\|}{|T_t|}$ along C_p, where $\|C_p\|$ is the length of the core path. Further, select $r = \frac{\|C_p\|}{|T_t|+1}$ such that the length of C_p is evenly divided between bounding sets, and overlap is minimized. The height h controls the maximum distance from the core path agents can move. The value of h selected will have an effect on the solution generated, as will the environment and obstacles, larger values allow agents to explore more. As we are interested in directed paths we select $h = 2\delta(N-1)$, which is the minimum non-overlapping distance for agents distributed along the height of B_k. It is important to note that rectangular bounding sets are *not* required for our work, they serve as a straightforward method of defining the position sample space in reference to the core path.

2) Constrained Bounding Set $\overline{B}_k$: Here we derive the constrained bounding sets $\overline{B}_k$, such that the induced interaction graph, $G_k(S_k)$, achieves our desired topological condition of completeness. Define the diameter of $\overline{B}_k$ as $d_{max} = \max\left\{\|x-y\| : \text{ for } (x,y) \in \overline{B_k} \subset \mathbb{R}^2\right\}$. Select $\overline{B_k}$ such that $d_{max} \leq \rho$, thus we achieve completeness of $G_k(S_k)$. We will assume equality to maximize coverage, i.e. $d_{max} = \rho$. The constrained bounding set should be a subset of the unconstrained, $\overline{B}_k \subseteq B_k$. To maximize objective (2), place the center of $\overline{B}_k$ at the region of *most interest* within B_k given sensing radius δ. Let $N_E^\rho : \mathbb{R}^2 \to \mathbb{R}^2$ be the ρ-radius neighborhood around a point. The center, $\mathbf{c}_k$, is selected via:

$$\underset{\mathbf{c}_k \in B_k \,:\, N_E^\rho(\mathbf{c}_k) \subset B_k}{\arg\max} \quad f_E(N_E^\rho(\mathbf{c}_k)) \tag{6}$$

We select $\mathbf{c}_k$ such that $\overline{B}_k$ does not intersect with obstacles. We calculate $\mathbf{c}_k$ by a discretization of B_k, ignoring points $< \rho$ from the boundary or obstacles, and evaluating $f_E(N_E^\rho(\mathbf{c}_k))$ for each point. Perfect selection of $\mathbf{c}_k$ is not necessary due to the sensing radii of agents. Given a limiting diameter of ρ, we select $\overline{B}_k = N_E^\rho(\mathbf{c}_k)$. When constrained we are forced to reduce coverage to meet the requirement, given that we maximize available coverage via our selection of $\overline{B}_k$. See Figure 1 for an example of bounding sets B_k and $\overline{B}_k$, the core path, agent paths, obstacles, and position samples.

B. Sample Generation

Within a B_k, we generate $|T_p|$ samples for each agent via a 2D uniform random distribution, while checking validity with a simple local planner. The local planner's purpose is to check that each sample has a direct path to the center of

bounding set, via an obstacle intersection calculation. Going back to the highway analogy, each position sample must have access to the highway, otherwise infeasible conditions can occur such as the sample being surrounded by an obstacle. The local planner used here is very similar to, and inspired by, the local planner used in many RRT-type algorithms [9], [23]. See Figure 1 for an example where the grey position samples do not appear in areas that are obscured/blocked by obstacles. Note that the local planner is a conservative solution, but it is quick to compute and is adaptable to complex motion models/configuration spaces [9].

Remark 1. Using the core path, we can generate bounding sets that guarantee feasible paths under the interaction constraint. The bounding sets, $B_{k\in T_t}$, allow for coverage increase, while remaining feasible due to the local planner and core path, and the bounding sets, $\overline{B}_{k\in T_t}$, allow agents to meet the interaction constraint.

C. Matroid Constraint

To select position samples, we need a *matroid constraint*. A matroid provides a general method of defining independence in combinatorial optimization [10], [27], [28]. A matroid $\mathcal{M}$ is an ordered pair $(\mathcal{V},\mathcal{I})$ consisting of a finite ground set $\mathcal{V}$ and a collection $\mathcal{I}$ of independent subsets of $\mathcal{V}$ satisfying three axioms as described in detail in [10]. We omit the axioms here for brevity. We make use of a uniform selection matroid so that we assign a single position per agent per time slice. Specifically, $\mathcal{M}_{u,k} = (\mathcal{V}_k, \mathcal{I}_{u,k})$ for each time $k \in T_t$. The independent sets are:

$$I_{u,k} = \{S \subset \mathcal{V}_k : |S_i \cap i| \leq 1, \forall i \in \mathcal{A}\} \tag{7}$$

where S_i is the set of agent indexes associated with samples in S. The constraint seems intuitive, but the rigor is necessary when considering approximation ratios. As we have $|T_t|$ time slices to consider, we define a matroid $\mathcal{M}_{u,k}$ $\forall k \in T_t$ because there are different ground sets, however the condition remains the same.

D. Sample Selection

We use a *greedy* algorithm to select samples for each agent. The greedy algorithm is well suited to submodular function maximization subject to a matroid constraint, as it guarantees a performance lower bound [10], [11], [26]. The greedy algorithm is as follows, begin with an empty set, add the element that maximizes *partial benefit*, continue until no additions are possible. Let the partial benefit from selecting element $e \in \mathcal{V}_k$, given an existing set $S_k \subseteq \mathcal{V}_k \setminus \{e\}$, for objective function g defined in (2), be:

$$\Delta_g(e|S_k) = g(S_k \cup \{e\}) - g(S_k) \tag{8}$$

For time slice $k \in T_t$, we initialize $S_0 = \{\emptyset\}$, and iteratively select the position $x_{ij}^{(k)} \in \mathcal{V}_k \setminus S_{n-1}$, that maximizes $\Delta_g(x_{ij}^{(k)}|S_{n-1})$ for step n, that is:

$$S_n = S_{n-1} \cup \left\{ \underset{x_{ij}^{(k)} \notin S_{n-1} : S_{n-1}\cup\{x_{ij}^{(k)}\}\in\mathcal{I}_{u,k}}{\arg\max} \Delta_g(x_{ij}^{(k)}|S_{n-1}) \right\} \tag{9}$$

Where $\mathcal{I}_{u,k}$ is defined in (7), and the algorithm terminates when $n = N$ giving $|S_k| = N$. Let a greedily constructed set be $S_k = \mathcal{G}(B_k \cap \mathcal{V}_k)$, meaning S_k is based on B_k and $\mathcal{V}_k$. Note that $B_k \cap \mathcal{V}_k$ indicates selection within bounding set B_k. Similarly, let an optimally constructed set be $S_k^* = \mathcal{O}(B_k \cap \mathcal{V}_k)$. That is, S_k^* is the best possible selection from $B_k \cap \mathcal{V}_k$. Optimal selections are generally not tractable [6], [10], but we use it for comparison.

IV. Time of Interaction Selection

Now we make the TOI selection of when to interact to meet the constraint. For all $k \in T_t$, we have the performances of the unconstrained and constrained greedy selections $g(\mathcal{G}(B_k \cap \mathcal{V}_k))$ and $g(\mathcal{G}(\overline{B}_k \cap \mathcal{V}_k))$, respectively. Similarly for the optimal case, we have selections $g(\mathcal{O}(B_k \cap \mathcal{V}_k))$ and $g(\mathcal{O}(\overline{B}_k \cap \mathcal{V}_k))$. We can begin by noting that:

$$\begin{aligned} g(\mathcal{G}(B_k \cap \mathcal{V}_k)) &\geq g(\mathcal{G}(\overline{B}_k \cap \mathcal{V}_k)) \\ g(\mathcal{O}(B_k \cap \mathcal{V}_k)) &\geq g(\mathcal{O}(\overline{B}_k \cap \mathcal{V}_k)) \end{aligned} \tag{10}$$

This is a consequence of $\overline{B}_k \subset B_k$, leading to more samples and possibly higher coverage. Further, by referencing [11], and by noting that (2) is a submodular function, under a matroid constraint, $\mathcal{M}_{u,k}$, we immediately get a lower bounded approximation ratio:

$$\begin{aligned} g(\mathcal{G}(B_k \cap \mathcal{V}_k)) &\geq \frac{1}{2} g(\mathcal{O}(B_k \cap \mathcal{V}_k)) \\ g(\mathcal{G}(\overline{B}_k \cap \mathcal{V}_k)) &\geq \frac{1}{2} g(\mathcal{O}(\overline{B}_k \cap \mathcal{V}_k)) \end{aligned} \tag{11}$$

When using greedy selection, we are guaranteed to perform at least half as well as the optimal case *per time slice*.

In order to make choices about the TOI, we need to express the difference in performance between the constrained and unconstrained cases, define:

$$\begin{aligned} L(T) &= \sum_{k\in T} g(\mathcal{G}(B_k \cap \mathcal{V}_k)) - g(\mathcal{G}(\overline{B}_k \cap \mathcal{V}_k)) \geq 0 \\ L^*(T) &= \sum_{k\in T} g(\mathcal{O}(B_k \cap \mathcal{V}_k)) - g(\mathcal{O}(\overline{B}_k \cap \mathcal{V}_k)) \geq 0 \end{aligned} \tag{12}$$

Where $L(T)$ and $L^*(T)$ are the sum of losses due to the constraint, for the times $T \subset T_t$. Given a set of times to interact $T \subset T_t$, with $T_t \setminus T$ being the non-interacting times, define a *total performance* function:

$$\begin{aligned} P(T) &= \sum_{k\in T} g(\mathcal{G}(\overline{B}_k \cap \mathcal{V}_k)) + \sum_{k\in T_t\setminus T} g(\mathcal{G}(B_k \cap \mathcal{V}_k)) \\ P^*(T) &= \sum_{k\in T} g(\mathcal{O}(\overline{B}_k \cap \mathcal{V}_k)) + \sum_{k\in T_t\setminus T} g(\mathcal{O}(B_k \cap \mathcal{V}_k)) \end{aligned} \tag{13}$$

where $P(T)$ and $P^*(T)$ are the total coverage for the greedy and optimal selections, respectively.

We claim that selecting $T \subset T_t$, with $|T| = K$ for some $K \in \{1, 2, \ldots, T_{end}\}$, such that $P(T)$ is maximized, is equivalent to picking T such that $L(T)$ is *minimized*. Minimizing the performance loss due to constraints will naturally *maximize* the total performance. Consider the following Lemma:

Lemma 1. Maximizing $P(T)$ is equivalent, in the optimal and greedy cases, to minimizing $L(T)$.

Proof. To see this, we have the following list of equivalent statements:

$$\begin{aligned}
&\underset{T\subset T_t,\,|T|\leq N}{\arg\max}\ P(T)\\
&=\underset{T\subset T_t,\,|T|\leq N}{\arg\min}\Big(\sum_{k\in T_t} g(\mathcal{G}(B_k\cap\mathcal{V}_k))-P(T)\Big)\\
&=\underset{T\subset T_t,\,|T|\leq N}{\arg\min}\Big(\sum_{k\in T_t} g(\mathcal{G}(B_k\cap\mathcal{V}_k))\\
&-\Big(\sum_{k\in T} g(\mathcal{G}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T} g(\mathcal{G}(B_k\cap\mathcal{V}_k))\Big)\Big)\\
&=\underset{T\subset T_t,\,|T|\leq N}{\arg\min}\Big(\sum_{k\in T} g(\mathcal{G}(B_k\cap\mathcal{V}_k))-\sum_{k\in T} g(\mathcal{G}(\overline{B}_k\cap\mathcal{V}_k))\Big)\\
&=\underset{T\subset T_t,\,|T|\leq N}{\arg\min}\ L(T)
\end{aligned} \tag{14}$$

Thus we maximize $P(T)$ via minimizing $L(T)$. Note that summing over all of T_t for a given B_k results in a fixed value. □

A key part of the coming result is that Assumption 1 holds. By definition, $L(T)$ is a modular performance function [26]. Minimization of a modular function, with a cardinality constraint, is known to be optimally-solved by the greedy algorithm [11]. The algorithm to generate T is as follows, evaluate $L(k)$ for each $k\in T_t$, sort into ascending order, and select the first K time slices. We need to generate unconstrained and constrained solutions to evaluate each $L(k)$.

Now for a lower bound on total performance using greedy selections. Let T_o and T_g be the set of times:

$$\begin{aligned}
T_o&=\underset{S\subset T_t:|S|=K}{\arg\max}\ L^*(S)\\
T_g&=\underset{S\subset T_t:|S|=K}{\arg\max}\ L(S)
\end{aligned} \tag{15}$$

Recall we refer to $P(T_g)$ as the *performance* of the greedy algorithm, and the set $\{S_k,\forall k\in T_g\}$ as the greedy *solution*. Similar nomenclature applies to the optimal case. Comparing performances yields the following result:

Theorem 1. The approximation ratio for the greedy solution to the interaction planning problem defined in Problem 1, compared to the optimal solution is $\frac{P(T_g)}{P^*(T_o)}=\frac{1}{2}$.

Proof.

$$\begin{aligned}
&\frac{P(T_g)}{P^*(T_o)}\\
&=\frac{\sum_{k\in T_g} g(\mathcal{G}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_g} g(\mathcal{G}(B_k\cap\mathcal{V}_k))}{\sum_{k\in T_o} g(\mathcal{O}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_o} g(\mathcal{O}(B_k\cap\mathcal{V}_k))}\\
&\geq\frac{\sum_{k\in T_o} g(\mathcal{G}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_o} g(\mathcal{G}(B_k\cap\mathcal{V}_k))}{\sum_{k\in T_o} g(\mathcal{O}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_o} g(\mathcal{O}(B_k\cap\mathcal{V}_k))}\\
&\geq\frac{\sum_{k\in T_o} \frac{1}{2}g(\mathcal{O}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_o} \frac{1}{2}g(\mathcal{O}(B_k\cap\mathcal{V}_k))}{\sum_{k\in T_o} g(\mathcal{O}(\overline{B}_k\cap\mathcal{V}_k))+\sum_{k\in T_t\setminus T_o} g(\mathcal{O}(B_k\cap\mathcal{V}_k))}\\
&=\frac{\frac{1}{2}P^*(T_o)}{P^*(T_o)}=\frac{1}{2}
\end{aligned} \tag{16}$$

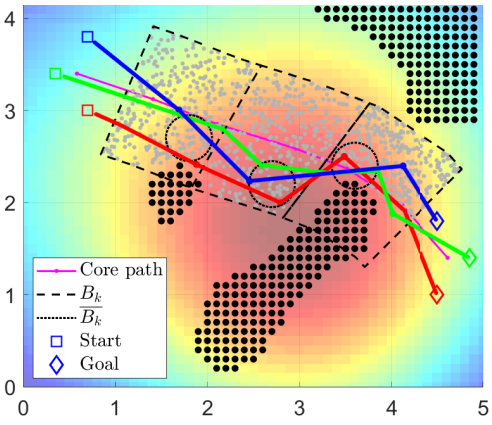


Fig. 1. Showing the core path C_p in purple, the bounding sets B_k as dashed black polygons, the bounding sets $\overline{B}_k$ as dashed black circles, the available position samples in grey, and obstacles in black. The environmental process is represented by the heatmap in the background, with red being higher value. Here $N=3$ and agent paths are shown in red, blue, and green, with squares as start locations and diamonds as goal locations.

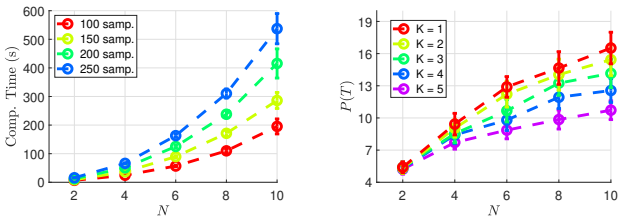
The first inequality is derived from T_g being selected so that $P(T_g)$ is *maximized*, any other set of K times has smaller total performance. The second inequality is from (11). □

Theorem 1 is a powerful result that allows us to quickly generate a solution to this problem which is otherwise intractable [10], [11], [16]. Using the proposed method allows for a much larger problem scale to be considered, while paying a small price in performance.

V. Simulations

Here we give simulations that verify the results derived. The simulations include a comparison to the method from [13] which solves a related problem. In addition, to achieve complete paths in our proposed solution, we use a MAS planning solution from [23], referred to as decentralized multi-agent RRT (DMA-RRT). Once samples are selected, we use DMA-RRT to plan paths for each agent from the start position, through samples avoiding obstacles and collisions, to the goal position. See Figure 1 for an example of DMA-RRT coupled with our method, where $K=1$ with $|T_t|=3$, the solution chosen has the second time slice to interact because it sacrifices the least coverage. Note that if there is a time difference between agents' arrivals at an *interacting* time slice, pauses in motion are introduced to account for it.

We compare with the work of [13] that solves a similar problem to ours, in the following we will refer to our method as intermittent interaction planning (IIP). Authors of [13], hereafter referred to as HS from the authors' initials, use a MIPP approach which *does not* include goal locations. To avoid introducing bias, we limit the path lengths between methods to be approximately equal per iteration. In addition, HS assumes the existence of a low level collision avoidance algorithm, where IIP explicitly deals with collisions, thus we omit the collision avoidance portion of IIP to match. Lastly, the interaction constraints are the same between solutions, these alterations allow us to fairly compare IIP with HS.



(a) (b)

Fig. 2. Showing in (a) the effect on computation time with increasing values of N at several values of $|T_p|$, and in (b) the effect on total performance with increasing values of N at several values of K. Each result is averaged for 120 fully randomized iterations, error bars correspond to one standard deviation.

The simulation parameters used are $\rho = 0.5$, and $\delta = 0.25$. The interval of disconnection, the distance agents can move before reconnecting in HS, was set to be 3 which limits the path lengths to compare. A square environment is used with $x_{\max} = y_{\max} = 5$, with *randomized* GMM parameters, obstacles, and starting positions, on each iteration, in this way environmental bias is removed. Obstacles are generated via smoothed perlin noise, cases without a solution are ignored. The environmental discretization parameter for HS is represented as e_x, here we use a rectangular grid of nodes for the environment graph though there are other possibilities [13].

In Figures 2 and 3 are Monte Carlo simulations which show computation time and total performance, $P(T)$, with *randomized* environmental conditions. In Figure 2 are two simulations that show computation time and total performance with increasing N for IIP only. For Figure 2a, with $|T_t| = 5$, $K = 2$, and for Figure 2b $|T_p| = 150$. Figure 2a shows the computation time of IIP with increasing N when $|T_p| = 100, \ldots, 250$. The computation time of IIP increases with N as well as with the size of the ground set, $\mathcal{V}_k$. Moreover, we are implicitly increasing $\mathcal{V}_k$ by increasing the number of agents, because the samples are taken for *each* agent. The ground set is of size $|\mathcal{V}_k| = N|T_p|$ thus it is natural that computation time would have the roughly N^2 scaling we see in Figure 2a. Furthermore, it follows that the computation time should scale linearly with increasing $|T_p|$, seen in Figure 2a as roughly uniform spacing between cases. In Figure 2b we show how the total performance of IIP changes with increasing N when $K = 1, \ldots, 5$. As the interaction constraint increases we get lower performance, with the worst performing case being $K = 5$, because we are forced to give up coverage in favor of interacting more often. Also note that with higher numbers of agents the performance becomes saturated in all cases, as expected with a coverage based objective function. The increase in error bar size with larger N, regarding Figures 2b and 3b, is due to the high number of variations that become possible with a large team of agents. There are simply more available options for a larger team in terms of paths chosen, interaction locations, and connected configurations.

Shown in Figure 3 are plots of computational time and total performance of both IIP and HS under increasing agent number and *position fidelity*, which is $|T_p|$ for IIP and e_x for HS. For Figure 3 we take $|T_t| = 3$ and $K = 1$ where HS follows the same interaction constraint (i.e. HS and IIP begin in a connected configuration, interact once along a path, and finish in a connected configuration). Readers are reminded that the time and sample parameters between Figures 2 and 3 are different. In Figure 3a, starting with computation time, the scaling of HS is expected, with a better represented environment the size of the path set also increases which heavily impacts computation time. For IIP, more samples having the same effect of better representing the environment, computation scales linearly as we are increasing the number of evaluations required for each agent by a constant multiple. The ranges of values for e_x and $|T_p|$ were selected to give roughly the same density of positions to each method. Now for total performance in Figure 3a, as expected HS outperforms IIP, though IIP respects and often outperforms the $\frac{1}{2}$ sub-optimality bound derived. The green line in Figure 3a is the difference between HS and IIP, provided for intuition on the $\frac{1}{2}$ bound. Total performance for each method increases slowly with increasing position fidelity, which follows intuition. The key concept of our IIP method, to scale tractably while maintaining bounded performance, is clearly seen in Figure 3a. As HS scales exponentially (right-hand side of Figure 3a top), IIP maintains comparatively low computational load while preserving guaranteed performance with HS. Taking into account the high computational cost of using HS, applications requiring fielded systems or edge computing would certainly benefit from using IIP in place of HS, especially when computational time is a priority (as in our previous search and rescue example).

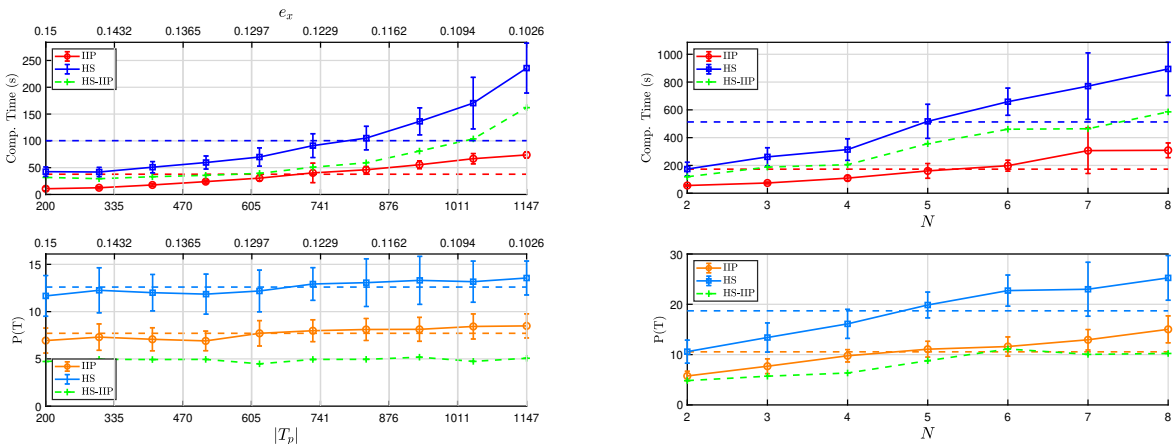
Shown in Figure 3b is a comparison between IIP and HS under increasing N. Here we take $e_x = 0.1$ and $|T_p| = 1200$. Since both methods theoretically scale linearly with N, and being informed by Figure 3a, we should expect a linear result from both methods with HS being slower than IIP. We see in Figure 3b, IIP having a smaller *per agent* gain in computation time compared to HS. For $P(T)$ comparison, in Figure 3b both methods exhibit near-linear scaling with increasing N, and IIP respects the bound derived in theory.

This work is accompanied by a repository where we have made code available at this Gitlab link[2] if others would like to simulate the proposed method.

VI. Conclusions

In this work we developed a method for environmental monitoring whilst intermittently interacting and avoiding obstacles. We accomplished this via a position sampling approach combined with greedy sample selection. For each time slice, we selected two position sets which allowed the times of interaction to be selected. Through this method, we achieved a performance ratio of $\frac{1}{2}$ compared to the optimal. We also presented several in-depth simulations to corroborate the results derived, and we compare to a method that uses iterative ascent in the path space. The Monte Carlo simulations showed the scalability of the proposed method under large agent and sample sets. In addition, the time required to compute the greedy solution compared with the optimal solution demonstrated the tractability of the proposed method.

[2]http://git.caslab.ece.vt.edu/hlarkin3/multi-agent-interaction-planning.git



(a) (b)

Fig. 3. In (a) showing a comparison between HS and IIP under increasing position fidelity, which is increasing $|T_p|$ for IIP and decreasing e_x for HS. The top plot shows computation time for both methods, while the bottom plot shows total performance for both methods, each data point is averaged for 24 iterations and the error bars represent ± 1 standard deviation. In (b) showing a comparison between HS and IIP under increasing N. The top and bottom plots show computation time and total performance, respectively, and here each data point is averaged for 18 iterations. The horizontal dashed lines represent the mean value across all data points, and the green dashed line represents the difference in value between HS and IIP.

REFERENCES

[1] T. Glotzbach, N. Crasta, and C. Ament, "Observability Analyses and Trajectory Planning for Tracking of an Underwater Robot using Empirical Gramians1," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 4215–4221, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016422639

[2] J. D. Quenzer and K. A. Morgansen, "Observability based control in range-only underwater vehicle localization," in *2014 American Control Conference*, Jun. 2014, pp. 4702–4707.

[3] L. Heintzman and R. K. Williams, "Nonlinear observability of unicycle multi-robot teams subject to nonuniform environmental disturbances," *Autonomous Robots*, vol. 44, no. 7, pp. 1149–1166, 2020.

[4] A. Bahr, J. J. Leonard, and M. F. Fallon, "Cooperative localization for autonomous underwater vehicles," *The International Journal of Robotics Research*, vol. 28, no. 6, pp. 714–728, 2009.

[5] N. Yu, H. Dai, A. X. Liu, and B. Tian, "Placement of connected wireless chargers," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 387–395.

[6] T.-W. Kuo, K. C.-J. Lin, and M.-J. Tsai, "Maximizing submodular set function with connectivity constraint: Theory and application to networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 2, pp. 533–546, 2015.

[7] L. Huang, J. Li, and Q. Shi, "Approximation algorithms for the connected sensor cover problem," in *International Computing and Combinatorics Conference*. Springer, 2015, pp. 183–196.

[8] R. K. Williams and G. S. Sukhatme, "Constrained interaction and coordination in proximity-limited multiagent systems," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 930–944, 2013.

[9] M. Svenstrup, T. Bak, and H. J. Andersen, "Trajectory planning for robots in dynamic human environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4293–4298.

[10] R. K. Williams, A. Gasparri, and G. Ulivi, "Decentralized matroid optimization for topology constraints in multi-robot allocation problems," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 293–300.

[11] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—i," *Mathematical programming*, vol. 14, no. 1, pp. 265–294, 1978.

[12] J. Liu and R. K. Williams, "Submodular optimization for coupled task allocation and intermittent deployment problems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3169–3176, 2019.

[13] G. Hollinger and S. Singh, "Multi-robot coordination with periodic connectivity," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 4457–4462.

[14] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4-5, pp. 432–442, 2011.

[15] Y. Kantaros, M. Guo, and M. M. Zavlanos, "Temporal logic task planning and intermittent connectivity control of mobile robot networks," *IEEE Transactions on Automatic Control*, 2019.

[16] J. Vondrák, "Submodularity in combinatorial optimization," 2007.

[17] G. A. Hollinger and S. Singh, "Multirobot coordination with periodic connectivity: Theory and experiments," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 967–973, 2012.

[18] J. Banfi, N. Basilico, and F. Amigoni, "Multirobot reconnection on graphs: Problem, complexity, and algorithms," *IEEE Transactions on Robotics*, no. 99, pp. 1–16, 2018.

[19] E. Stump, N. Michael, V. Kumar, and V. Isler, "Visibility-based deployment of robot formations for communication maintenance," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4498–4505.

[20] J. Banfi, N. Basilico, and S. Carpin, "Optimal redeployment of multirobot teams for communication maintenance," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3757–3764.

[21] D. Tateo, J. Banfi, A. Riva, F. Amigoni, and A. Bonarini, "Multiagent connected path planning: Pspace-completeness and how to deal with it," in *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI2018)*, 2018, pp. 4735–4742.

[22] R. K. Williams, A. Gasparri, A. Priolo, and G. S. Sukhatme, "Distributed combinatorial rigidity control in multi-agent networks," in *52nd IEEE Conference on Decision and Control*. IEEE, 2013, pp. 6061–6066.

[23] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4956–4961.

[24] R. K. Williams and G. S. Sukhatme, "Observability in topology-constrained multi-robot target tracking," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1795–1801.

[25] J. Liu and R. K. Williams, "Optimal intermittent deployment and sensor selection for environmental sensing with multi-robot teams," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1078–1083.

[26] L. Lovász, "Submodular functions and convexity," in *Mathematical Programming The State of the Art*. Springer, 1983, pp. 235–257.

[27] J. Oxley, "On the interplay between graphs and matroids," *LONDON MATHEMATICAL SOCIETY LECTURE NOTE SERIES*, pp. 199–239, 2001.

[28] I. Shames and T. H. Summers, "Rigid network design via submodular set function optimization," *IEEE Transactions on Network Science and Engineering*, vol. 2, no. 3, pp. 84–96, 2015.