Resilient Coverage: Exploring the Local-to-Global Trade-off

Ragesh K. Ramachandran[1], Lifeng Zhou[2], James A. Preiss[1] and Gaurav S. Sukhatme[1]

Abstract— We propose a centralized control framework to select suitable robots from a heterogeneous pool and place them at appropriate locations to monitor a region for events of interest. In the event of a robot failure, our framework repositions robots in a user-defined local neighborhood of the failed robot to compensate for the coverage loss. If repositioning robots locally fails to attain a user-specified level of desired coverage, the central controller augments the team with additional robots from the pool. The size of the local neighborhood around the failed robot and the desired coverage over the region are two objectives that can be varied to achieve a user-specified balance. We investigate the trade-off between the coverage compensation achieved through local repositioning and the computation required to plan the new robot locations. We also study the relationship between the size of the local neighborhood and the number of additional robots added to the team for a given user-specified level of desired coverage. Through extensive simulations and an experiment with a team of seven quadrotors we verify the effectiveness of our framework. We show that to reach a high level of coverage in a neighborhood with a large robot population, it is more efficient to enlarge the neighborhood size, instead of adding additional robots and repositioning them.

I. Introduction

Real-world applications of robots require resilience. A multi-robot team holds the promise of resilience in a failure-prone environment [1]. That is because, if some robots in the team fail, the remaining ones can continue the task by reconfiguration via replanning [1], [2]. For example, in a multi-robot coverage problem, in the event of a robot failure, the neighbors of a failed robot can reposition themselves to fill the coverage gap induced by the failed robot [1]. Alternately, when a resource, such as a sensor or a computation unit on a robot fails, the robot team repositions itself to reconfigure the communication network to maintain the availability of resources among the robots in the team [2].

In practice, the resilience objective depends on specific task(s) that the robots are performing. For example, in an urgent, time-critical task, such as using robots to fight fire [3] or deliver medical supplies [4], the failures need to be handled quickly. In tasks such as mine mapping [5] and plant monitoring [6], when some robots fail, safety and performance are more important even if it takes longer for the system to reconfigure in response to a failure. In this paper, we propose a framework that provides a user with the
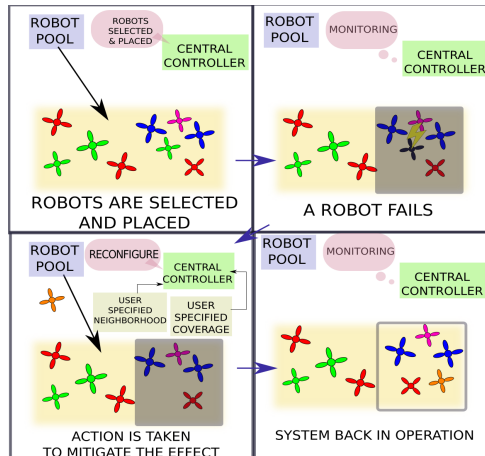

Fig. 1: An illustration of our framework: Local vs. global tradeoff decision making governs whether to locally reconfigure or to replenish from a pool.

flexibility to choose different resilience objectives, thereby resulting in the most preferable resilient actions suitable for the task.

We consider a scenario in which a central decision-maker or controller selects an appropriate set of robots from a collection of heterogeneous robots to perform a monitoring task. The robots are selected based on various criteria such as their reliability, cost of deployment and size of the sensing area. The selected robot team is placed in the environment such that the coverage of the robot team over the environment is maximized. The coverage of a robot team (defined more precisely in Section II-B) over a domain is the probability that the robot team detects some event of interest in the domain (e.g. intrusion of adversarial agents in the domain). Coverage also quantifies the monitoring performance of the robot team over the domain. While performing the monitoring task, it is possible that some robots may fail to operate. Consequently, the central controller needs to decide whether it is possible to monitor the environment at an acceptable performance level by (1), team reconfiguration (i.e. repositioning the remaining (active) robots) or (2), by providing additional robots to raise the monitoring performance of the team to an acceptable level. The central controller makes this decision based on parameters set by a user. Figure 1 schematically illustrates of our framework.

Our framework is as follows. If a robot fails, the user selects a neighborhood of size $L \in \mathbb{R}^+$ (an *L-neighborhood*) around the failed robot. For ease of computation, we use a square which circumscribes a circle of radius L as the user-defined L-neighborhood. In formal terms, the L-neighborhood is a ball of size L centered around the failed

[1]Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA (email: {rageshku,japreiss, gaurav}@usc.edu).
[2]Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA (email: lfzhou@vt.edu).

robot in the ∞-norm. Active (non-failed) robots within the L-neighborhood are then required to reposition to achieve the desired level of coverage demanded by the user. The coverage requirement is expressed by a single parameter $\gamma \in [0, 1]$ which is defined as the ratio of the coverage attained by the robots after repositioning to the coverage before the robot failure occured in the L-neighborhood.

If the robots inside the L-neighborhood do not attain the user specified coverage level by repositioning themselves, then the central controller augments the robot team with a new set of robots to sufficiently meet the user's demand. Notably, a smaller L would result in faster reconfiguration, but may not contribute enough to the coverage demand if the γ value is high. The parameters L and γ help the user to trade off between the coverage attained and the reconfiguration efficiency when dealing with robot failures. We investigate this trade-off in this paper.

Related work. Multi-robot coverage is a well studied topic in robotics [7]–[12]. Following the seminal paper [7], most approaches focus on constructing an objective function commonly called a coverage functional. It quantifies the coverage achieved by a robot team and corresponding strategies which optimize the coverage functional. Although, significant research has gone into defining good coverage functionals, even extending the idea to non-Euclidean spaces [10], most are non-convex and the solutions are only local optima. Recently, researchers have started to exploit the diminishing returns property of some functions (submodularity [13]) for obtaining $1 - 1/e$ suboptimal solutions using greedy strategies [13]–[15]. Similar to the works [14], [15], in this paper, we use a greedy algorithm to achieve suboptimal coverage over the environment by the robots.

Resilience in multi-robot systems is an active area of research [1], [16]–[18]. The approach in [1] focuses on building resilient strategies for multi-robot sweep coverage problems. In contrast, here we introduce resilient strategies to handle robot failures in blanket coverage problems. Section II-B describes the blanket coverage problem in detail. In [16]–[18], the authors design robust strategies for task execution assuming at most a certain fixed number of robots would fail during the execution of a task. Whereas, in our paper, we consider if some robot fails, how to actively coordinate the remaining robots to compensate for the loss from the failure. We present our framework as four different problems and our overall strategy as solutions to these problems.

Notation. We use $\mathbb{P}(\cdot)$ to represent the probability of occurrence of an event. For any positive integer $Z \in \mathbb{N}^+$, $[Z]$ denotes the set $\{1, 2, \cdots, Z\}$. We use $\mathbb{R}_+$, $\mathbb{R}_{\geq 0}$ and $\mathbb{R}^d$ to denote positive real numbers, non-negative real numbers, and d-dimensional real vectors with $d \in \mathbb{N}^+$, respectively. The vector of ones is represented as $\mathbf{1}$. For any countable set $\mathcal{X}$, $|\mathcal{X}|$ denotes its cardinality and $2^{\mathcal{X}}$ is its collection of subsets or power set. A non-negative set function $f : 2^{\mathcal{X}} \to \mathbb{R}_{\geq 0}$ is said to be submodular if $\forall \mathcal{A} \subseteq \mathcal{B} \subset \mathcal{X}$ and $\forall x \in \mathcal{X} \backslash \mathcal{B}$, $f(\mathcal{A} \cup \{x\}) - f(\mathcal{A}) \geq f(\mathcal{B} \cup \{x\}) - f(\mathcal{B})$. The condition implies that the function has a diminishing returns

property. Moreover, the function is monotone if $\forall \mathcal{A} \subseteq \mathcal{B}$, $f(\mathcal{A}) \leq f(\mathcal{B})$ and normalized if $f(\emptyset) = 0$.

II. PROBLEM FORMULATION

Consider a pool of $N \in \mathbb{N}^+$ heterogeneous robots labeled $\{1, 2, \cdots, N\}$ with varying deployment costs and sensing ranges. A central controller is tasked with selecting robots from the pool and employing the selected robots to efficiently monitor a compact area of interest. We refer to this area of interest as $\mathcal{Q} \subset \mathbb{R}^2$. The central controller monitors the activities of the selected team and takes appropriate measures in the event of a robot failure.

The framework in this paper is presented as solutions to four problems: (1) initial team selection (2) global placement, (3) local reconfiguration and (4) intermediate robot selection. *Initial team selection* deals with the problem of selecting the smallest set of robots from the robot pool to perform the task of monitoring $\mathcal{Q}$ while satisfying constraints on the cost of deployment and reliability. Once a team is selected, it is optimally placed in the region of interest ($\mathcal{Q}$) so that the region can be monitored for events of interest. The solution to the *global placement* problem dictates the optimal locations for placing the robot team in $\mathcal{Q}$. In the event of a robot failure, based on the user specified parameters L and γ, the central controller tries to compensate for loss in monitoring performance by repositioning the active robots in a local neighborhood of the failed robot (specified by L). We refer to this as the *local reconfiguration* problem. As the user can specify a desired level of local coverage through the parameter γ, if the local reconfiguration does not yield the level of local coverage demanded by the user, then the central controller selects and places additional robots from the robot pool such that the demanded local coverage is satisfied. Since this robot selection problem occurs during the task execution, we call it the *intermediate robot selection* problem. We formalize these problems in the forthcoming subsections. Our solutions to these problems are detailed in Section III.

A. Initial team selection

As mentioned earlier, a central controller is tasked with the process of selecting a team of robots that meet certain requirements from the robot pool. One requirement is the reliability of the robots in the selected team. We describe the probability that a robot will operate successfully for a given time period $[T_0, T]$ by using a reliability function [19]. The reliability function for robot $i \in [N]$ is defined as, $R_i(T_0, T) = \exp\{-\int_{T_0}^{T} \lambda_i(\tau)d\tau\}$, where $\lambda_i(\tau)$ is the instantaneous rate of failure of the robot i due to various factors, e.g., battery drain, processor failure, actuator failure, etc. It is reasonable to assume that $\lambda_i(\tau)$ is a non-decreasing function. We approximate the bathtub model [19] of $\lambda_i(\tau)$ with a quadratic function. Specifically, we define $\lambda_i(\tau) = \lambda_{i0} + k_i\tau^2$. The parameters of this function are computed through data fitting [19]. Let $\boldsymbol{t}_i^f$ denote the random variable representing the failure time of robot i. According to our

reliability model, the failure time of the robot is described as,

$$\mathbb{P}(\boldsymbol{t}_i^f \in [T_0, T]) = 1 - R_i(T_0, T). \tag{1}$$

Therefore, if we assume that robot failures are independent events, then the probability that a set of $\mathcal{S} \in 2^{[N]}$ robots fail during the time interval $[T_0, T]$ can be computed as $\prod_{i \in \mathcal{S}}(1 - R_i(T_0, T))$.

Suppose a particular task requires a set of J distinct resources for its successful execution and $\bar{\gamma}_j$ is the minimum amount of resource $j \in [J]$ required by the team to perform the task. Let $\Gamma_j : 2^{[N]} \to \mathbb{R}_+$ be a mapping which maps a robot team to the amount of resource j the team possesses as a whole. In this paper, we assume the Γ_j has an additive structure, namely,

$$\Gamma_j(\mathcal{S}) = \sum_{i \in \mathcal{S}} \Gamma_{ij}, \tag{2}$$

where $\mathcal{S} \in 2^{[N]}$ and $\Gamma_{ij} \geq 0$ is the amount of resource j maintained by robot i. In addition, we define $C(\mathcal{S})$ as the cost of deploying the robot team $\mathcal{S}$. Next we present the first problem in this paper.

Problem 1 (**Initial team selection**). *Find the minimum number of robots from a pool, such that the selected robot team is equipped with sufficient resources to monitor a region of interest under the constraints that the total cost of deployment is below a budget β and the probability that all the robots fail before the monitoring task is completed is below a threshold $\alpha \in [0, 1]$. More formally,*

$$\min_{\mathcal{S} \in 2^{[N]}} |\mathcal{S}| \tag{3}$$

subject to,

$$C(\mathcal{S}) \leq \beta \tag{4}$$

$$\prod_{i \in \mathcal{S}}(1 - R_i(0, T)) \leq \alpha, \ T > 0 \tag{5}$$

$$\sum_{i \in \mathcal{S}} \Gamma_{ij} \geq \bar{\gamma}_j \ \forall j \in [J] \tag{6}$$

where T is the time for which the robots are entrusted with the monitoring task.

For ease of computation, we adopt the following structure for the budget constraint (Equation 4):

$$C(\mathcal{S}) = \sum_{i \in \mathcal{S}} C_i \leq \beta, \ C_i > 0, \tag{7}$$

where C_i is the cost of deploying robot i. The cost of deployment captures the cost incurred during its deployment due to fueling and other overheads. Note that, the availability of a robot in the pool may not mean that it has zero cost of deployment. Since in this paper, the task under consideration is coverage, we substitute the resource constraint (Equation 6) with the following constraint,

$$\sum_{i \in \mathcal{S}} \langle \mathcal{A}_i \rangle \geq \delta \langle \mathcal{A}_\mathcal{Q} \rangle, \ \delta > 1, \tag{8}$$

where $\langle \mathcal{A}_i \rangle$ is the sensing area associated with robot i and $\langle \mathcal{A}_\mathcal{Q} \rangle$ is the area of $\mathcal{Q}$. If a_i is the finite sensing range of robot i, then its sensing region is $\mathcal{A}_i = \{\boldsymbol{p} : \|\boldsymbol{x}_i - \boldsymbol{p}\| \leq a_i\}$. δ is a predefined parameter which determines the amount of redundancy in the selected robot team in terms of the total area that the team can ideally cover.

Once a robot team is selected for the task, the central controller has to assign appropriate locations to robots in the selected team such that the coverage over the area of interest is maximized. In the next subsection, we formally define the notion of coverage used in this paper and mathematically describe our robot placement problem.

B. Global placement

Our formulation of the multi-robot domain coverage problem is similar to the ones presented in [7], [14]. After obtaining a robot team $\mathcal{S}$ with $|\mathcal{S}| \leq N$ by solving Problem 1, the coverage problem deals with placing these robots such that the probability of detecting events of interest (e.g. arrival of an adversarial target in the area) over an area is maximized.

We proceed by formulating our coverage problem as a blanket coverage problem [20]. The map $\phi : \mathcal{Q} \to \mathbb{R}_+$ represents a probability density function which quantifies the amount of information or probability of occurrence of an event of interest over $\mathcal{Q}$. In other words, the probability that an event of interest occurred in the region $\mathcal{D} \subseteq \mathcal{Q}$ can be written as:

$$\mathbb{P}(\boldsymbol{e} \in \mathcal{D}) = \int_\mathcal{D} \phi(\boldsymbol{p})\mathrm{d}\boldsymbol{p} \tag{9}$$

where $\mathbb{P}(\boldsymbol{e} \in \mathcal{D})$ represents the probability that an event of interest occurred in $\mathcal{D}$. Consequently, $\phi(\cdot)$ has a bounded support $\mathcal{Q}$ and $\int_\mathcal{Q} \phi(\boldsymbol{p})\mathrm{d}\boldsymbol{p} = 1$. Denote the set containing the positions of robot set $\mathcal{S}$ as $\mathcal{X}_{|\mathcal{S}|} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_{|\mathcal{S}|}\}$, where $\boldsymbol{x}_i \in \mathbb{R}^2$. The sensing performance of robot i at a point $\boldsymbol{p}$ is a non-increasing function of the distance between $\boldsymbol{p}$ and $\boldsymbol{x}_i$ [7]. Consequently, we define the sensing performance function of the robot $i \in \mathcal{S}$ as $s_i : \mathbb{R}_{\geq 0} \to (0, 1]$ as a non-increasing function of $\|\boldsymbol{x}_i - \boldsymbol{p}\|$. An example of such a function which is used for the simulations in this paper is

$$s_i(\|\boldsymbol{x}_i - \boldsymbol{p}\|) = \exp(-\eta_i\|\boldsymbol{x}_i - \boldsymbol{p}\|), \tag{10}$$

where η_i is the sensing decay rate of robot i. Let the random variable $\boldsymbol{d}_i^p \in \{0, 1\}$ model the event of detecting an event of interest which occurred at $\boldsymbol{p} \in \mathcal{Q}$ by robot i, then

$$\mathbb{P}(\boldsymbol{d}_i^p = 1|\boldsymbol{x}_i, \boldsymbol{p}) = \begin{cases} s_i(\|\boldsymbol{x}_i - \boldsymbol{p}\|) & \text{if } \boldsymbol{p} \in \mathcal{A}^i \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

We use $\mathbb{P}(\boldsymbol{d}_i^{\boldsymbol{p}})$ instead of $\mathbb{P}(\boldsymbol{d}_i^p = 1|\boldsymbol{x}_i, \boldsymbol{p})$ for brevity. If we assume that the detection probabilities of different robots are independent, then the probability of detection of an event of interest by the robot team, given that the event occurred at a point $\boldsymbol{p}$, can be computed as,

$$\mathbb{P}(\cup_i \boldsymbol{d}_i^p = 1|\mathcal{X}_{|\mathcal{S}|}, \boldsymbol{p}) = 1 - \prod_{i \in \mathcal{S}}[1 - \mathbb{P}(\boldsymbol{d}_i^{\boldsymbol{p}})]. \tag{12}$$

Therefore, the probability of detecting events of interest in $\mathcal{Q}$ by a team of robots, $\mathcal{S}$, is

$$H(\mathcal{X}_{|\mathcal{S}|}, \mathcal{Q}) = \int_{\mathcal{Q}} \mathbb{P}(\cup_i \boldsymbol{d}_i^{\boldsymbol{p}} = 1 | \mathcal{X}_{|\mathcal{S}|}, \boldsymbol{p})\phi(\boldsymbol{p})\mathbf{d}\boldsymbol{p}. \quad (13)$$

Note that this quantity is a measure of the coverage attained by the robots over the environment $\mathcal{Q}$. Also, the coverage functional value quantifies the monitoring performance of the robot team over $\mathcal{Q}$. We formally define this robot team placement problem as,

Problem 2 (**Global placement**). *Given a team of robots, $\mathcal{S}$, compute their positions $\mathcal{X}_{|\mathcal{S}|} \subseteq \mathcal{Q}$ such that the coverage functional $H(\mathcal{X}_{|\mathcal{S}|}, \mathcal{Q})$ is maximized.*

After the robots, $\mathcal{S}$, reach their appropriate locations in $\mathcal{Q}$ dictated by the solution to Problem 2, the central controller enters a monitoring mode. It monitors the activities of the robot team and initiates actions that enable the robot team to maintain a user-specified level of coverage when a robot in the team fails.

C. Resilient coordination to robot failure

After the global placement, the robot team starts monitoring the environment. As time goes by, robots may fail due to internal and/or external factors, such as the battery drain or the adversarial attack. We model this failure by Equation 1. If a robot fails, it loses its monitoring functionality and stops contributing to the coverage functional $H(\mathcal{X}_{|\mathcal{S}|})$. The central controller employs the standard heart beat signal mechanism [1, Section 4.1] to detect the failure of a robot. The central controller listens to a periodic heart beat signal send by each robot and assumes a robot has failed if the controller did not receive the robot's heart beat signal within a pre-defined time interval. We denote the failed robot by r_f and the remaining active robots in the team by $\mathcal{S} \setminus r_f$. We seek to design a resilient strategy that can react and compensate for the loss from a robot failure by reconfiguring the robots which lie in a local neighborhood around the failed robot. Formally,

Problem 3 (**Local reconfiguration**). *Given that a robot failed, reposition the robots in its local neighborhood to mitigate the coverage loss caused by the robot failure. We defer the details of this problem to Section III-C.*

If the robots in a user-defined local neighborhood around the failed robot fail to provide the desired level of coverage by reconfiguring, then the central controller selects robots from the robot pool and deploys them to achieve the desired coverage level. The problem can be described as,

Problem 4 (**Intermediate robot selection**).

$$\min_{\mathcal{S}^{\text{new}} \in 2^{[\hat{N}]}} |\mathcal{S}^{\text{new}}| \quad (14)$$

subject to,

$$\prod_{i \in \mathcal{S}^{\text{new}}} (1 - R_i(0, T - T_f)) \leq \alpha(T_f), \ 0 < T_f < T \quad (15)$$

$$\sum_{i \in \mathcal{S}^{\text{new}}} \langle \mathcal{A}_i \rangle \geq \langle \mathcal{A}_f \rangle \quad (16)$$

where $\mathcal{S}^{\text{new}}$ is the newly selected robot set and $[\hat{N}]$ is the current set of robots in the robot pool. $\alpha(T_f) = \frac{\alpha}{\prod_{j \in \mathcal{S} \setminus r_f}(1 - R_j(T_f, T))}$ with T_f denoting the time instant when the failure happens. Note that, $\alpha(T_f)$ is the maximum probability of failure corresponding to the newly selected robots such that total probability of failure of the new team, $\mathcal{S}^{\text{new}} \bigcup \{\mathcal{S} \setminus r_f\}$ (newly selected robots and the active robots), is less than α. $\langle \mathcal{A}_f \rangle$ is the sensing area of the failed robot.

III. METHODOLOGY

In this section, we describe in detail the strategies adopted by the central controller to solve the problems presented in Section II.

A. Robot selection strategies

Recall that we consider two types of robot selection problems in our framework: (1) initial team selection (Problem 1) and (2) intermediate robot selection (Problem 4). Mathematically, both robot selection problems can be cast as *mixed integer linear programming* (MILP) [21].

Initial team selection solution. We first describe the MILP associated with Problem 1. Let the vector $\boldsymbol{\Pi} := [\pi_1, \pi_2, \cdots, \pi_N]^\top$ denote the collection of binary decision variables which encode the robot team selection. Then the following MILP formally describes Problem 1 as,

$$\underset{\boldsymbol{\Pi} \in \{0,1\}^N}{\text{minimize}} \quad \mathbf{1}^\top \boldsymbol{\Pi} \quad (17)$$

$$\text{subject to} \quad [C_1, C_2, \cdots, C_N]^\top \boldsymbol{\Pi} \leq \beta \quad (18)$$

$$\left[\tilde{R}_1, \tilde{R}_2, \cdots, \tilde{R}_N\right]^\top \boldsymbol{\Pi} \leq \tilde{\alpha}, \quad (19)$$

$$[\langle \mathcal{A}_1 \rangle, \langle \mathcal{A}_2 \rangle, \cdots, \langle \mathcal{A}_N \rangle]^\top \boldsymbol{\Pi} \geq \delta \langle \mathcal{A}_{\mathcal{Q}} \rangle, \quad (20)$$

where $\tilde{R}_i = \log(1 - R_i(0, T))$ and $\tilde{\alpha} = \log(\alpha)$. Solving the MILP above yields the solution to Problem 1.

Intermediate robot selection solution. Since Problem 4 is similar in formulation to Problem 1, it can be solved using a similar MILP. Let $\boldsymbol{\Pi}^p$ be vector of length N such that its i-th element is 1 if the robot with label i exists in the robot pool, otherwise 0. Then the MILP formulation for Problem 4 can be written as:

$$\min_{\boldsymbol{\Pi} \in \{0,1\}^N} \quad \mathbf{1}^\top \boldsymbol{\Pi} \quad (21)$$

$$\left[\tilde{R}_1, \tilde{R}_2, \cdots, \tilde{R}_N\right]^\top \boldsymbol{\Pi} \leq \tilde{\alpha}(T_f), \quad (22)$$

$$[\langle \mathcal{A}_1 \rangle, \langle \mathcal{A}_2 \rangle, \cdots, \langle \mathcal{A}_N \rangle]^\top \boldsymbol{\Pi} \geq \langle \mathcal{A}_f \rangle, \quad (23)$$

$$\boldsymbol{\Pi} \leq \boldsymbol{\Pi}^p. \quad (24)$$

Equation 24 ensures that only the robots currently available in the robot pool can be selected. The inequality is defined element-wise. Solving the MILP above results in the selection of a minimum set of robots from the the robot pool satisfying the conditions in Problem 4.

Algorithm 1: Greedy Placement

Input:
- Robot team $\mathcal{S}$ and ground location set $\hat{\mathcal{Q}}$;
- Monitoring function $H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{Q}})$.

Output: Placement set $\mathcal{X}_{|\mathcal{S}|}$.

1: $\mathcal{X}_{|\mathcal{S}|} \leftarrow \emptyset$
2: **while** $|\mathcal{X}_{|\mathcal{S}|}| < N$ **do**
3: $\quad s = \underset{s \in \hat{\mathcal{Q}} \setminus \mathcal{X}_{|\mathcal{S}|}}{\operatorname{argmax}} H(\mathcal{X}_{|\mathcal{S}|} \cup \{s\}, \hat{\mathcal{Q}}) - H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{Q}})$
4: $\quad \mathcal{X}_{|\mathcal{S}|} \leftarrow \mathcal{X}_{|\mathcal{S}|} \cup \{s\}$
5: **end while**

B. Initial placement

After solving Problem 1, the central controller selects an initial robot team $\mathcal{S}$. The selected robot team is then tasked to monitor the environment, which we formally describe in Section II-B as Problem 2. The objective is to maximize the total coverage functional $H(\mathcal{X}_{|\mathcal{S}|}, \mathcal{Q})$ by finding a set of $|\mathcal{S}|$ placement locations $\mathcal{X}_{|\mathcal{S}|}$. Similar to [14], if we finely discretize $\mathcal{Q}$ into a finite collection of $W \in \mathbb{N}^+$ grid cells as $\hat{\mathcal{Q}} = \{\hat{Q}_1, \hat{Q}_2, \cdots, \hat{Q}_W\}$, then the optimal coverage problem using at most $|\mathcal{S}|$ robots can be expressed as

$$\max_{\mathcal{X}_{|\mathcal{S}|} \in \hat{\mathcal{Q}}} H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{Q}}) \qquad (25)$$
$$\text{s.t. } |\mathcal{X}_{|\mathcal{S}|}| \leq |\mathcal{S}| \ll \hat{\mathcal{Q}}, \hat{\mathcal{Q}} \subseteq \mathcal{Q}.$$

where $H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{Q}})$ is the coverage function defined on the discretized environment $\hat{\mathcal{Q}}$.

It turns out that the discrete form of the coverage functional in Equation 25 is a *normalized monotone submodular function* and therefore has the diminishing returns property [13], [14], [16]. The diminishing returns property captures the notion that the more robots participate in a monitoring task, the less gain one gets by adding an extra robot towards the task. Also, in Problem 2, we consider a cardinality constraint, i.e., $|\mathcal{X}_{|\mathcal{S}|}| \leq |\mathcal{S}|$. That is, we can place the robots in no more than $|\mathcal{S}|$ locations. This is because the robot team only has $|\mathcal{S}|$ robots in total.

The maximization of submodular functions under a cardinality constraint is generally NP-hard [13], [14]. However, a simple greedy algorithm that selects an element with the maximal marginal gain on the submodular function in each round can give a constant-factor $(1 - 1/e)$ approximation of the optimal [13]. We use the greedy algorithm [13] for solving the monitoring problem (Problem 2) in Algorithm 1. The running time of Algorithm 1 can be bounded as $\mathcal{O}(|\mathcal{S}|^2 W)$.

C. Tunable resilient coordination

In this section, we describe the resilient coordination that the central controller employs to mitigate the effect of robot failures on the monitoring task. Our resilient coordination is a combination of the solutions to both Problem 3 and Problem 4. We refer to this combined solution as tunable resilient coordination.

Algorithm 2: Tunable Resilient Coordination

1: $\mathcal{X}_f^L \leftarrow \emptyset$
2: $[\mathcal{X}_f^L, \sim] = $ Algorithm $1(\mathcal{R}_f^L, \hat{\mathcal{N}}_L)$
3: $\mathcal{X}_{|\mathcal{S} \setminus r_f|} = \mathcal{X}_f^L \cup \mathcal{X}_f^O$
4: **if** $H(\mathcal{X}_{|\mathcal{S} \setminus r_f|}, \hat{\mathcal{N}}_L)/H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{N}}_L) \geq \gamma$ **then**
5: $\quad$ Terminate
6: **else**
7: $\quad$ Request a new robot set $\mathcal{S}^{\text{new}}$ by solving the MILP in Problem 4
8: $\quad \mathcal{R}_f^L = \mathcal{R}_f^L \cup \mathcal{S}^{\text{new}}$
9: $\quad [\mathcal{X}_f^L, \sim] = $ Algorithm $1(\mathcal{R}_f^L, \hat{\mathcal{N}}_L)$
10: $\quad \mathcal{X}_{|\mathcal{S} \setminus r_f|} = \mathcal{X}_f^L \cup \mathcal{X}_f^O$
11: **end if**

In the first step, the robots in a neighborhood of the failed robot coordinate to counter the coverage loss due to the robot failure (solution to Problem 3). For the failed robot r_f, we refer to the robots inside a neighborhood of size L around it as its *L-neighbors* and denote them as $\mathcal{R}_f^L$. Also, we use $\mathcal{N}^L$ to represent the *L-neighborhood*. $\hat{\mathcal{N}}_L \subseteq \hat{\mathcal{Q}}$ is the set of grid cells in $\hat{\mathcal{Q}}$ contained in $\mathcal{N}^L$. We denote the robots outside the L-neighborhood of the failed robot r_f as $\mathcal{R}_f^O$. Notably, the neighborhood size L is a user-defined tuning parameter, which measures the extent to which a user wants to utilize the currently active robots in the team. We first reposition the failed robot's L-neighbors $(\mathcal{R}_f^L)$ using Algorithm 1 to counter the coverage loss through a local repositioning $\mathcal{R}_f^L$ inside $\hat{\mathcal{N}}_L$. This yields a new set of positions $\mathcal{X}_f^L$ for $\mathcal{R}_f^L$ (Algorithm 2, line 2). It is noteworthy that, after this step, only the positions of $\mathcal{R}_f^L$ change. The positions of $\mathcal{R}_f^O$, denoted $\mathcal{X}_f^O$, remain unaltered. Thus, we obtain a new configuration for the active robots, denoted by $\mathcal{X}_{|\mathcal{S} \setminus r_f|}$ ($\mathcal{X}_f^L \cup \mathcal{X}_f^O$, Algorithm 2, line 3). We denote the coverage of the current robot team ($\mathcal{S} \setminus r_f$, active robots) within L-neighborhood by $H(\mathcal{X}_{|\mathcal{S} \setminus r_f|}, \hat{\mathcal{N}}_L)$.

In the second step, we compare the coverage of the new configuration, $H(\mathcal{X}_{|\mathcal{S} \setminus r_f|}, \hat{\mathcal{N}}_L)$, with that of the configuration before the robot failure, $H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{N}}_L)$, in the L-neighborhood. We denote $\gamma \in [0, 1]$ as a user-defined ratio. If $H(\mathcal{X}_{|\mathcal{S} \setminus r_f|}, \hat{\mathcal{N}}_L)/H(\mathcal{X}_{|\mathcal{S}|}, \hat{\mathcal{N}}_L) \geq \gamma$, the new configuration is good enough to compensate for the loss and the resilient coordination is done (Algorithm 2, lines 4-5). Otherwise, the central controller solves the MILP (Problem 4) to select a new robot set from the robot pool (Algorithm 2, line 7), inserts the new robot set into L-neighborhood of the failed robot (Algorithm 2, line 8), redoes the local repositioning (Algorithm 2, line 9) and obtains the configuration for the new team (Algorithm 2, line 10).

Recall that the parameter γ encodes the desired coverage level required by the user. For example, if γ is set to 1, then the user forces the central controller to adopt a strategy that maintains the same level of coverage over the L-neighborhood as before (without the robot failure). A higher value of γ demands a higher level of coverage and possibly

more new robots from the robot pool. In future, we will perform a formal analysis on how the solution of Problem 1 affects the solutions to the other problems.

IV. SIMULATION

We validate our framework for resilient coverage through simulations based on Problems 1-4 and their corresponding solutions in Section III.

Simulation setup. We generate a pool of 50 heterogeneous robots with mean-life span: 420, and a standard deviation that is 10% of the mean. The reliability function parameters for each robot are calculated based on its life span [19]. The sensing area and the cost of a robot are generated proportionally to its life span. We assume that the most expensive robot costs 50 and has a sensing area of 200. We set the parameters in the MILP (Equation 17) as $\beta = 500$, $\alpha = 0.3$ and $\delta = 1$. We select a robot team from a pool of 50 heterogeneous robots to cover a square domain of size 30×30 during a time period, 500. Thus, we set the area of the domain as $\langle \mathcal{A}_Q \rangle = 900$.

We place the robots in the square domain based on the positions generated from Algorithm 1. The robots are tasked with monitoring the square domain for a time period of 500.

To simulate a robot failure, we choose a random time during $0 \sim 500$. At the chosen random time, we pick the failed robot based on the reliability value using a roulette wheel technique [22].

Results. Figure 2 depicts three random failure instants and the repositioning of the active robots after a robot failure. Here, we set $\gamma = 1$. In all cases, the L-neighbors of the failed robot (black dot) try to compensate the coverage loss induced by the failed robot through reconfiguration.

To investigate the trade-off between the size of L-neighborhood and the coverage attained by repositioning L-neighbors without adding new robots from the pool, we simulate 10 trials of random robot failure. In each trial, we compute the coverage functional value attained by the reconfigured robots for different values of $L \in \{10, 15, 20\}$. The results are presented in Figure 3a. Also, Figure 3b shows the computational time required by Algorithm 1 for different values of L. From Figure 3, it is clear that there exists a trade-off between the coverage value attained through local repositioning of L-neighbors and the computational time required to compute the reconfiguration. Therefore, a user can utilize this trade-off to choose a preferred neighborhood size, L.

Moreover, we simulate another 10 trials of robot failure to quantify the number of new robots added to the local L-neighbors with $\gamma = 1$ in Figure 4. Figure 4 shows that the number of new robots requested decreases when the L-neighborhood size increases, which is intuitively true. Comparing the results in Figure 3 and Figure 4 one could arrive at the wrong conclusion that adding new robots to achieve the desired coverage is equivalent to increasing L. We use Figure 5 to throw some light on this misconception.

Again, we simulate 10 trials of robot failure. In each trial, a failed robot is chosen randomly and its L-neighbors are locally reconfigured. We refer to the coverage functional value of this configuration as the *base coverage*. Then using Algorithm 1, we add new sets of robots with sizes $\{10, 20, 30, 40\}$ in the L-neighborhood in terms of different L values, $L \in \{10, 15, 20\}$. The coverage values and the percentage increase of the coverage over the base coverage is shown in Figure 5. Figure 5 shows that for a small L ($L = 10$) the change in percentage increase by adding 40 robots as compared to 10 robots is roughly $\approx 5\%$, while this change is roughly tripled ($\approx 15\%$) when $L = 20$. One possible reason can be that the coverage functional is submodular with the diminishing returns property. This means that adding more robots may not substantially increase the coverage when a large number of robots are already placed in a small neighborhood. Therefore, as a rule of thumb, we propose that it is better to use a small γ with a small L if the user needs to compensate efficiently for a coverage loss. However, if a user wants to reach a high coverage, i.e., γ is large, enlarging the size of L-neighborhood can be necessary.

V. EXPERIMENT

We demonstrate the practical application of our approach on a hardware system with seven quadrotors. Our experiment is built around a real-time implementation of Algorithm 2 with $\gamma = 0.0$. By setting $\gamma = 0.0$, the condition in Algorithm 2, line 4 is always true, ensuring that no new robots are needed to be deployed to achieve the desired coverage. This is done purely for the ease of experiment implementation. After taking off and moving to a coverage formation, we simulate a sequence of hardware failures in the robot team. When a failure occurs, the affected robot immediately begins a landing trajectory. Meanwhile, the system queries a human operator to select the neighborhood size L around the failed robot. The user is presented with a graphical interface displaying the robots' positions and an illustration of the neighborhood perimeter, allowing them to choose the trade-off between coverage and repositioning cost based on the current scenario.

Real hardware experiments require a multi-robot motion planning algorithm to generate collision-free trajectories between the initial and repositioned placements. Since motion planning is not the main focus of this paper, we make several assumptions to simplify this subproblem. First, we assume a homogeneous robot team such that each goal position can be filled by any robot in the team. Second, we assume that the robots are of negligible size compared to their distances in the coverage formation. These two assumptions allow a simple motion planning solution using a goal assignment that minimizes the sum of distances between each robot's start position and goal position, and following straight-line trajectories. Such an assignment cannot have intersecting trajectories except for degenerate cases [23]. Our implementation verifies that the trajectories do not pass too closely to account for the size of the robots. In the general case of heterogeneous robots, or when the robot sizes

(a) Initial placement with $L = 5$ (b) Initial placement with $L = 7.5$ (c) Initial placement with $L = 10$

(d) Repositioning with $L = 5$ (e) Repositioning with $L = 7.5$ (f) Repositioning with $L = 10$
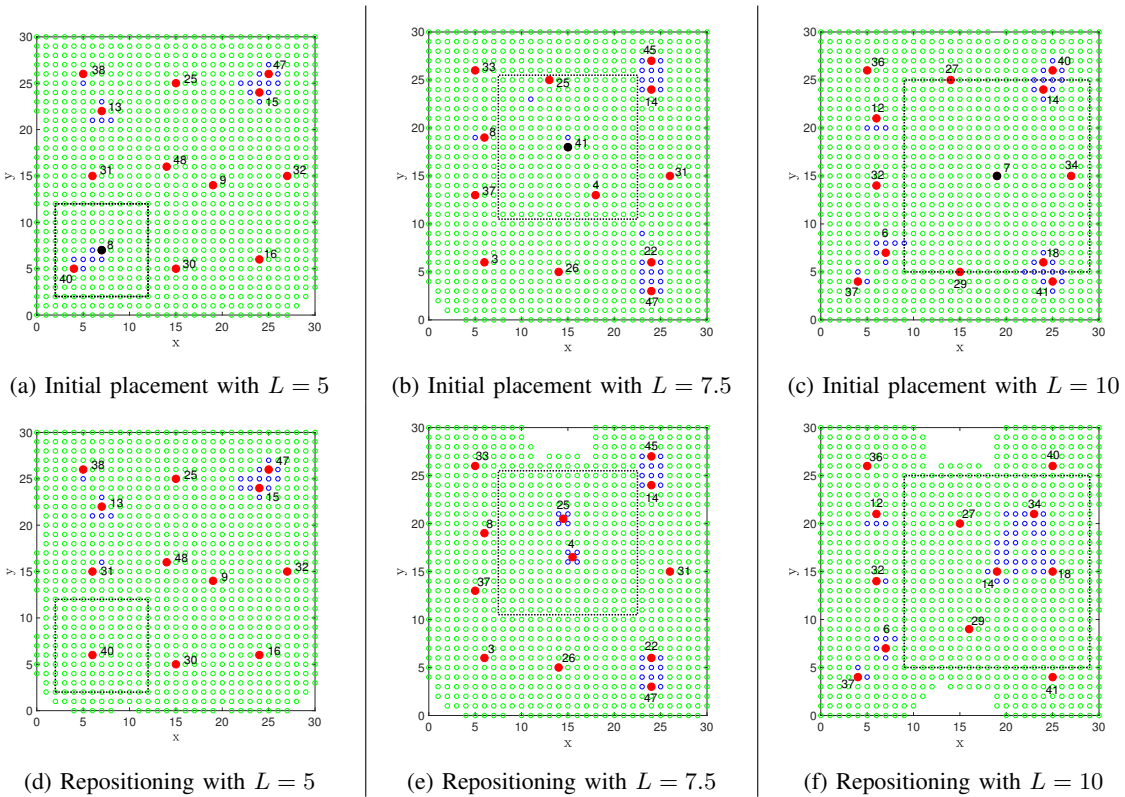
Fig. 2: The figures illustrate three simulated failure instants. The failed robots are colored in black and the active robots are colored in red. The numbers indicate the labels of the robots selected from a pool of 50 robots. The inner dotted square depicts the local neighborhood. Blue circles indicate regions of high coverage value (≥ 0.97), green circles shows the regions with coverage value between 0.50 and 0.97.
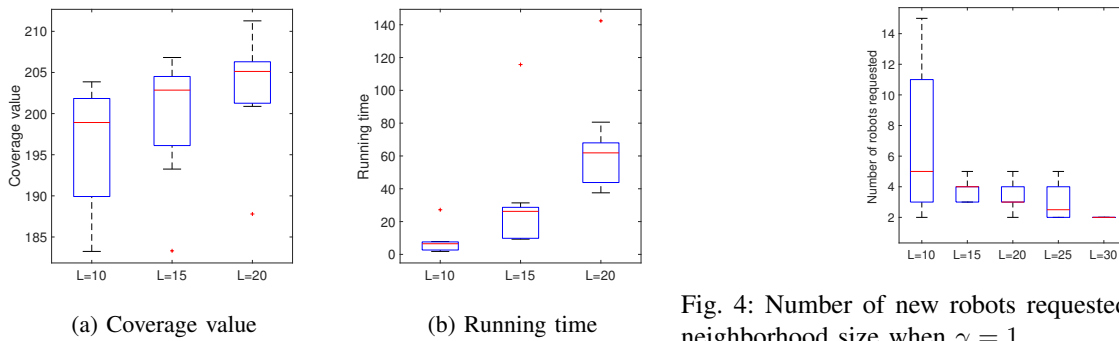

(a) Coverage value   (b) Running time

Fig. 3: (a) Comparison of the coverage over the domain for different neighborhood sizes L. (b) The computational time (in seconds) required to find the locations of the robots in the neighborhood for different neighborhood sizes.

are large compared to the formation size, a more advanced motion planning algorithm such as [24] can be used.

Our experiment is implemented on the Crazyswarm platform [25] composed of miniature quadrotors in an indoor motion capture space, shown in Figure 6. After the operator chooses the value of L, we generate a new configuration using Algorithm 2 and solve the optimal assignment problem using the Hungarian algorithm [26]. Each robot receives its new goal position over the radio and executes a smooth straight-line trajectory using onboard polynomial trajectory planning, sensor fusion, and control. A video of this exper-
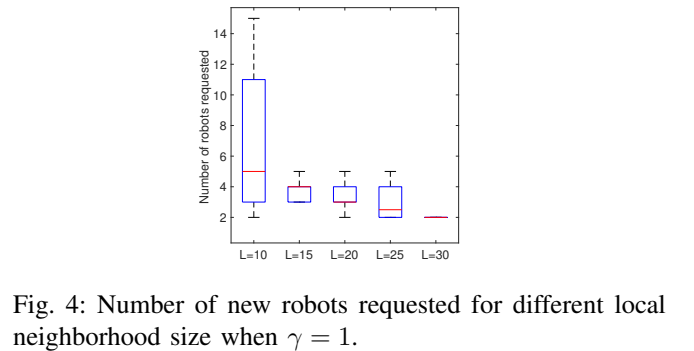

Fig. 4: Number of new robots requested for different local neighborhood size when $\gamma = 1$.

iment is available in the supplementary material.

VI. CONCLUSION

This article presents a novel centralized framework for robot team selection and placement in a region such that the coverage over the region is maximized. Additionally, the framework provides a resilient coordination strategy to handle robot failures during a monitoring task. In particular, the framework rearranges the robots in a user-specified local neighborhood around the failed robot to attain a user-defined coverage level. If local repositioning does not achieve the desired coverage, the framework augments the robot team with new robots from a pool to meet the coverage demanded by the user. By specifying the size of the local neighborhood and the desired coverage level, the user can trade off the

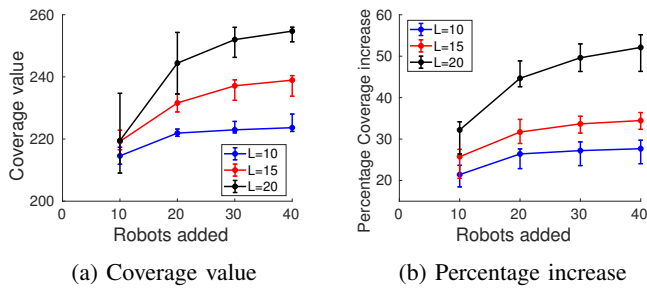(a) Coverage value   (b) Percentage increase

Fig. 5: (a) Variation in coverage value of the team over the domain, when robots are added to a failed robot's neighbor and repositioned within the neighborhood, for different neighborhood sizes L. (b) Percentage increase in the coverage value over the domain for the data in Figure 5a.
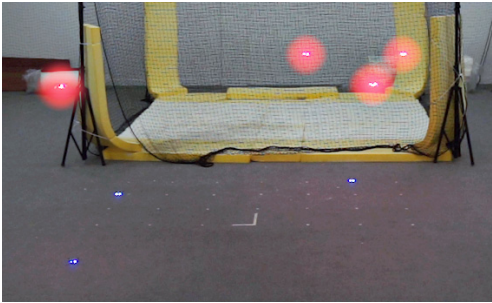

Fig. 6: Demonstration of Algorithm 2 with seven quadrotors. Quadrotors highlighted with a red circle are currently active. Three quadrotors have suffered hardware failures and landed. Experimental details in Section V.

amount of coverage attained and the computational time required to achieve the desired coverage level.

Our framework is validated through simulations and a proof-of-concept experiment using a team of seven quadrotors. From the simulation results, we infer that adding more robots to the robot team may not always result in sufficient increase in the coverage due to the diminishing returns property of the coverage function.

We are currently working on rigorously analyzing the interplay between user-defined parameters and the coverage performance. A future avenue is to incorporate decentralized submodular optimization [27] into our framework and study the trade-off between centralized and decentralized components in the framework.

REFERENCES

[1] J. Song and S. Gupta, "Care: Cooperative autonomy for resilience and efficiency of robot teams for complete coverage of unknown environments under robot failures," *Autonomous Robots*, pp. 1–25, 2019.

[2] R. K. Ramachandran, J. A. Preiss, and G. S. Sukhatme, "Resilience by reconfiguration: Exploiting heterogeneity in robot teams," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 6518–6525.

[3] K. Harikumar, J. Senthilnath, and S. Sundaram, "Multi-uav oxyrrhis marina-inspired search and dynamic formation control for forest firefighting," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, pp. 863–873, 2018.

[4] E. Ackerman and E. Strickland, "Medical delivery drones take flight in east africa," *IEEE Spectrum*, vol. 55, no. 1, pp. 34–35, 2018.

[5] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 3. IEEE, 2003, pp. 4270–4275.

[6] U. Weiss and P. Biber, "Plant detection and mapping for agricultural robots using a 3d lidar sensor," *Robotics and autonomous systems*, vol. 59, no. 5, pp. 265–273, 2011.

[7] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[8] M. Schwager, F. Bullo, D. Skelly, and D. Rus, "A ladybug exploration strategy for distributed adaptive coverage control," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 2346–2353.

[9] M. A. Batalin and G. S. Sukhatme, "Spreading out: A local approach to multi-robot coverage," in *Distributed Autonomous Robotic Systems 5*. Springer, 2002, pp. 373–382.

[10] S. Bhattacharya, R. Ghrist, and V. Kumar, "Multi-robot coverage and exploration on riemannian manifolds with boundaries," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 113–137, 2014.

[11] M. A. Batalin and G. S. Sukhatme, "The analysis of an efficient algorithm for robot coverage and exploration based on sensor network deployment," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 3478–3485.

[12] S. Poduri and G. S. Sukhatme, "Constrained coverage for mobile sensor networks," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, 2004, pp. 165–171 Vol.1.

[13] 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.

[14] X. Sun, C. G. Cassandras, and X. Meng, "Exploiting submodularity to quantify near-optimality in multi-agent coverage problems," *Automatica*, vol. 100, pp. 349 – 359, 2019.

[15] X. Sun, C. G. Cassandras, and K. Gokbayrak, "Escaping local optima in a class of multi-agent distributed optimization problems: A boosting function approach," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 3701–3706.

[16] L. Zhou, V. Tzoumas, G. J. Pappas, and P. Tokekar, "Resilient active target tracking with multiple robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 129–136, 2018.

[17] K. Saulnier, D. Saldana, A. Prorok, G. J. Pappas, and V. Kumar, "Resilient flocking for mobile robot teams," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1039–1046, 2017.

[18] L. Guerrero-Bonilla, A. Prorok, and V. Kumar, "Formations for resilient robot teams," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 841–848, 2017.

[19] S. B. Stancliff, J. M. Dolan, and A. Trebi-Ollennu, "Mission reliability estimation for repairable robot teams," *International Journal of Advanced Robotic Systems*, vol. 3, no. 2, p. 24, 2006.

[20] A. V. Savkin, T. M. Cheng, Z. Xi, F. Javed, A. S. Matveev, and H. Nguyen, *Optimal Distributed Blanket Coverage Problem*. John Wiley & Sons, Ltd, 2015, ch. 6, pp. 87–95.

[21] A. Schrijver, *Theory of Linear and Integer Programming*. New York, NY, USA: John Wiley & Sons, Inc., 1986.

[22] T. Back, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[23] M. Turpin, N. Michael, and V. Kumar, "Capt: Concurrent assignment and planning of trajectories for multiple robots," *I. J. Robotics Res.*, vol. 33, no. 1, pp. 98–112, 2014.

[24] W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Trans. Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[25] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *ICRA*. IEEE, 2017, pp. 3299–3304.

[26] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

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