國 立 中 央 大 學

數學研究所

碩士論文

Multi-robot Search in 3D Environments using Submodularity with Matroid Intersection Constraints

研究生: 李晏碩

指導教授: 曾國師

中 華 民 國 一百一十三 年 六 月

國 立 中 央 大 學

數學研究所

碩士論文

Multi-robot Search in 3D Environments using Submodularity with Matroid Intersection Constraints

研究生: 李晏碩

指導教授: 曾國師

中 華 民 國 一百一十三 年 六 月

Multi-robot Search in 3D Environments using Submodularity with Matroid Intersection Constraints

摘要

研究生: 李晏碩

指導教授: 曾國師

關鍵字: 次模性, 擬陣理論, 多機器人搜尋問題, 任務分配問題

多機器人搜尋是一個具有挑戰性的問題，因爲其涉及任務分配和覆蓋問題，而這些問題皆是NP-hard。 它可以重新定義爲在擬陣限制下的覆蓋率最大化問題。 覆蓋率最大化問題可透過次模性來解決。擬陣限制是由路徑限制和分群限制所組成。 此研究提出Multi-robot Search with Matroid constraints (MRSM)的方法，此方法達成$\frac{1}{3}\widetilde{OPT}$，其中 $\widetilde{OPT}$ 是基於生成樹結構下的近似最優性能。 實驗結果顯示，所提出MRSM方法在多機器人搜尋問題中優於其他演算法。

Multi-robot Search in 3D Environments using Submodularity with Matroid Intersection Constraints

Abstract

Author: Yan-Shuo, Li

Adivisor: Kuo-Shih, Tseng

Keywords: Submodularity, Matroid, Multi-robot search problem, Task allocation problem

The multi-robot search problem is challenging since it involves task allocation and coverage problems, which are NP-hard. This problem is reformulated as the maximal coverage problem subject to the intersection of matroid constraints. The coverage problem is solved by utilizing its submodularity. The intersection matroid is composed of a routing constraint and a clustering constraint. The proposed algorithm, Multi-robot Search with Matroid constraints (MRSM), achieves $\frac{1}{3}\widetilde{OPT}$, where $\widetilde{OPT}$ is an approximately optimal performance under a spanning tree structure. The experiment results show that the proposed approach outperforms state-of-the-art methods in multi-robot search problems.

Contents

Figures

Tables

1 Introduction

Multi-robot search is a widely studied topic due to emerging application domains such as target tracking [4], search and rescue [5], and industrial inspection [6]. The multi-robot search problem involves coverage, multi-robot task allocation, and routing problems. Challenges of these problems are as follows. First, finding the maximal coverage under a budget is computational intractability [7]. Second, the task allocation to multiple robots is another issue [8]. Third, each robot in multi-robot search problems needs to solve the traveling salesman problem (TSP) [9]. Solving these problems is NP-hard.

Recent approaches to solving these problems are as follows. For the multi-robot search problem, the researchers propose an efficient path planning algorithm (eMIP) [10]. The problem is solved by a sequential single robot path planning algorithm. Furthermore, each robot needs to solve the TSP, which results in poor time complexity. A theoretical performance guarantee is provided. However, the search performance deteriorates as the number of robots increases. The problem is reformulated as submodular maximization subject to intersection system constraints (MRSIS) [1]. The search problem is solved by generating a set of robot trajectories while considering the balance of task assignments. However, it requires solving the minimum spanning tree (MST) problem, and the theoretical performance is not provided.

For multi-robot task allocation problems, Markov Decision Processes (MDP) over graphs are adopted. In [2], the researchers propose a reinforcement learning algorithm based on a graph neural network. The task assignment problem is solved by selecting the node with the highest probability for each robot. However, the routing problem and workload balance are not considered.

For search via learning approaches, the researchers propose a prob-

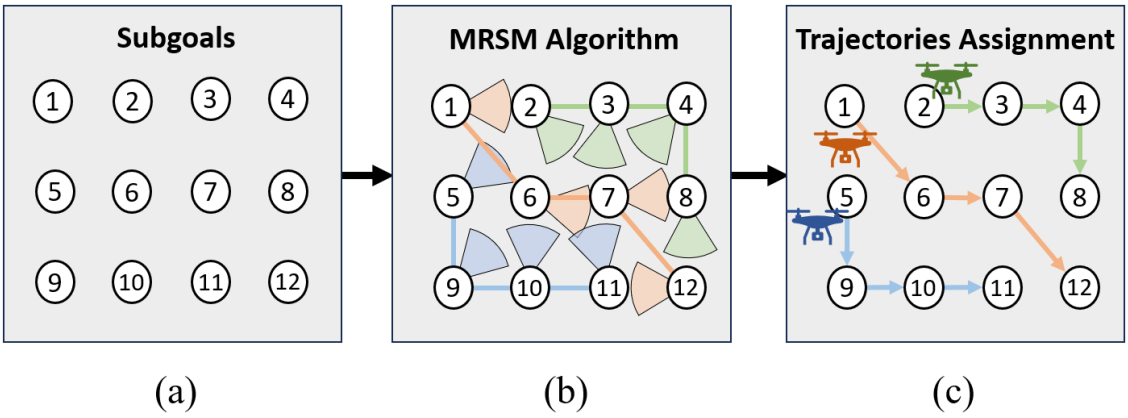


Figure 1.1: Overview of the proposed multi-robot search system with three robots. (a) An example of subgoals (nodes) in a search environment. (b) The trajectories for each robot (orange, blue, and green lines) are obtained by the proposed algorithm (MRSM). The circular sectors are the coverage areas. (c) The trajectories are assigned to robots to find targets in the environment.

ability density function (PDF) as a reward function that generates a sequence of decisions based on the reinforcement learning method [3]. The search problem and the allocation problem are solved by selecting actions with the maximal individual reward function for each robot. Yet, the graph only considers the unit cost for routing, and the balance of the robots' workloads is not taken into account, potentially resulting in poor task assignment.

To resolve the aforementioned issues, this research proposes a Multi-Robot Search with Matroid constraints (MRSM) algorithm[1], which is an improved version of MRSIS algorithm [1]. Maximal coverage and balanced workloads are considered simultaneously. The coverage function is to cover a larger area for robots, while the balancing function is to measure the equality of the workloads assigned to robots. Besides, the routing constraint is reformulated as matroid to boost the theoretical guarantees of the MRSIS [1].

The proposed MSRM method is illustrated in Fig. 1.1. Given an environment map with subgoals, the goal is to find all targets with multiple robots as soon as possible. In Fig. 1.1(a), subgoals are evenly distributed in the space. A weighted graph $\mathcal{G}(V, E, w)$ is constructed, where V represents the subgoal set, E represents the edge set, and w is an Euclidean distance function. In Fig. 1.1(b), the MSRM generates a

[1] The primary distinction between MRSIS and MRSM is that MRSIS models the clustering and routing constraints as independence systems while MRSM models two constraints as matroids.

set of trajectories that maximize the environment coverage and maintain balanced workloads among robots. In Fig. 1.1(c), trajectories are then assigned to robots to search for targets in the environment.

In this research, some assumptions are made. First, MRSM relies on known environments. A set of subgoals is evenly distributed in the search environments. Second, the coverage at every subgoal can be pre-computed since the search method is based on known maps. Third, the perception of robots includes uncertainty and targets may be occluded in the environment.

The contributions of this research are as follows. First, the submodular maximization under matroid intersection constraints (MRSM) is proposed for multi-robot search problems. To the best of our knowledge, this is the first work to propose this objective for these problems. Second, thanks to the submodularity, the theoretical guarantees of MRSIS [1] and MRSM are proved as $\frac{1}{2+k_G}\overline{OPT}$ and $\frac{1}{3}\widetilde{OPT}$, respectively, where $k_G > 1$, $\overline{OPT}$ is an approximately optimal solution of the MST or TSP solver and $\widetilde{OPT}$ is an approximately optimal solution of the spanning tree. Notice that the theoretical bound of MRSM does not depend on the number of robots. Third, the experiment results show that the proposed method (MRSM) outperforms state-of-the-art approaches (e.g., MRSIS [1], CapAM [2] and PD-FAC [3]) in the multi-robot search problem.

This paper is organized as follows. Section 2 reviews the relevant work on target search methods, multi-robot task allocation, and routing constraints. Section 3 describes the background knowledge of this research. Section 4 introduces the problem formulation. Section 5 describes the search algorithm. Section 6 describes the experiments and analyzes the results. Finally, Section 7 draws conclusions and outlines future work.

1.1 Publication Note

Portions of the literature survey, problem formulation, and experiments appeared in [1], [11], [12].

2 Related Work

In this section, recent work on target search methods, multi-robot task allocation, and routing constraints is reviewed.

2.1 Target Search

Searching for targets in an environment is a sequential decision-making problem. Therefore, search methods based on the type of decision-making algorithm, such as submodular approaches, the frontier-based search, the next-best-view search, the probabilistic search, and Partially Observable Markov Decision Processes (POMDP), can be adopted to generate the search strategy.

The submodular method formulates the target search problem as submodular maximization under a resource constraint. Singh et al. [10] introduce an efficient path planning algorithm (eMIP), which coordinates multiple robots to obtain highly informative paths subject to the robot's path cost. The eMIP finds solutions that achieve at least $\frac{1-1/e}{1+\log_2 N}$, where N is the number of robots. However, the performance deteriorates when more robots are involved. Li et al. [1] propose a Multi-Robot Search with an Intersection System (MRSIS) algorithm. The objective is to generate a set of robot trajectories that maximize a coverage function and balance robot workloads under clustering and routing cost constraints. The clustering and routing constraints are formulated as an intersection system.

The frontier-based search method utilizes the frontier between the explored and unexplored space to expand existing knowledge. Zhou et al. [13] propose a hierarchical framework, Fast UAV ExpLoration (FUEL), that supports UAV exploration in complex unknown environments. It improves the exploration efficiency using the incremental frontier information structure. The work is extended to decentralized multi-

UAV [14]. Instead of allocating frontiers and viewpoints to the UAVs, the task assignment is based on an online hierarchical decomposition to ensure that all UAVs simultaneously explore distinct regions.

The next-best-view (NBV) planning determines the next viewpoint that provides the most valuable information to improve search efficiency. Mittal et al. [15] adopt NBV planner for exploration and propose a novel landing site detection algorithm that computes costmaps based on several hazard factors (e.g., terrain flatness, steepness, depth accuracy, and energy consumption information). Lauri et al. [16] propose a submodular utility function for multi-sensor NBV planning under partition matroid constraints. In addition, the utility function coordinates view selection and prevents overlapping views among multiple sensors.

The probabilistic approach is to estimate the target location under sensor and target motion uncertainty. Robots make decisions based on the probability distribution. Mohamed et al. [17] present the person search-orienting problem (PSOP). It adopts the user activity probability density functions (APDFs) to generate a search plan to maximize the expected target detection with the limited search time. The approach is then extended to multiple robots by generating a team plan to cooperate effectively [18]. Sheng et al. [3] propose a probability density factorized (PD-FAC) multi-agent distributional reinforcement learning method that decomposes the PDF of the multi-robot system into a set of individual value distributions. It is guaranteed that the objective function of the overall system's value distribution can be linearly approximated by the same reliability metric defined over the agent's individual value distribution.

Formulating robotic search problems as Partially Observable Markov Decision Processes (POMDP) has recently become a popular method. POMDP considers the search problem where the states of target locations and robot sensors are uncertain. A sequence of actions is generated by maximizing a reward function. Zhu et al. [19] propose a Dec-POMDP method to find a target in an environment with obstacles. The approach provides a scalable framework for a large number of UAVs. It enables the UAV swarm to cooperate efficiently by sharing limited observations in

the mission. Many methods constrain the search space in 2D. Zheng et al. [20] present a multi-object search method, 3D Multi-Object Search (3D-MOS), in 3D environments with a frustum-shaped field of view, which can be applied to mobile robots or drones.

2.2 Multi-Robot Task Allocation (MRTA)

The goal of MRTA is to optimize an objective function within a given budget for multiple robots. However, finding an optimal solution is NP-hard [8].

Several research studies have employed submodular maximization with matroid constraints to solve the MRTA problem. This problem involves various challenges, such as the orienteering problem [21], the intermittent deployment problem [22], and the capacitated vehicle routing problem [23]. If the objective function is submodular, greedy algorithms can find solutions with theoretical guarantees [24]. In the team surviving orienteers (TSO) problem [25], an independent set of a matroid is selected for maximizing the expected visited nodes at least one robot. These nodes also ensure that each vehicle reaches its destination with probabilities above a specified threshold. In environmental monitoring application [26], the multi-robot task allocation problem and the multi-robot intermittent deployment problem are formulated as submodular maximization with matroid intersection constraints. In the surveillance task allocation in urban environments [27], a decentralized algorithm, which applies auction methods to assign tasks with matroid constraints, is proposed.

Recent research on multi-robot task allocation has been paying attention to deep reinforcement learning approaches with graph neural network (GNN) [28]. Paul et al. [2] propose a Capsule Attention-based Mechanism (CapAM), a graph reinforcement learning architecture that encodes graph information such as robot location, task deadline, and elapsed mission time. The approach can be easily scaled to a large number of tasks. Tolstaya et al. [28] develop a GNN architecture with behavior cloning to solve the coverage problem. Using a sparse representation of the local graph connectivity, the approach can scale to larger maps and teams. Zhang et al. [29] propose a Hierarchical-Hops Graph

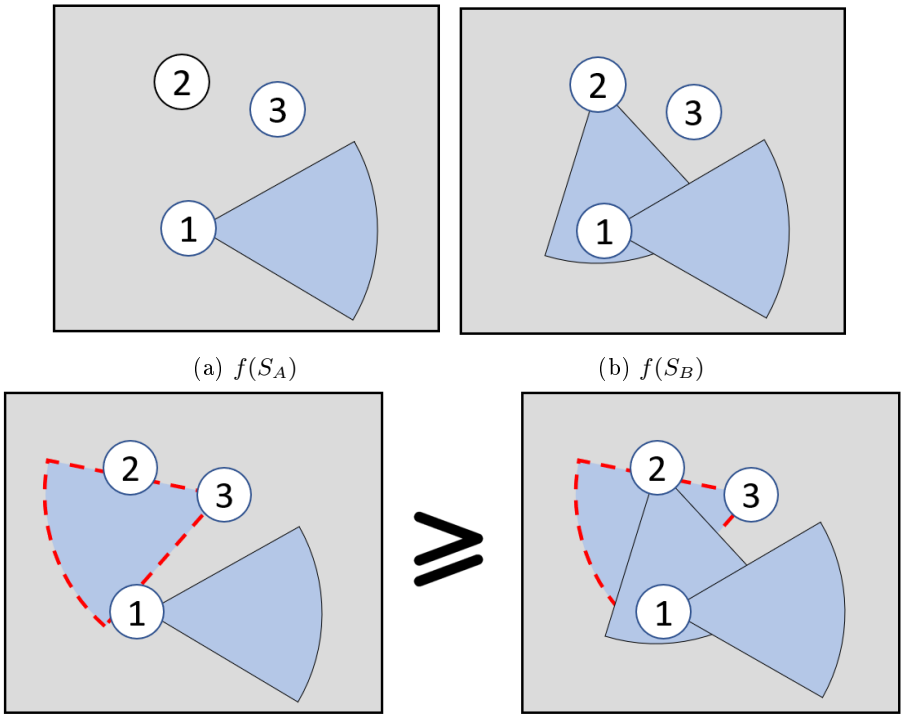
Neural Networks (H2GNN), which evaluates the importance of graph nodes at different hops through the multi-head attention mechanism. To improve exploration efficiency, multi-agent reinforcement learning is applied to learn collaborative strategies.

2.3 Routing Constraints

Submodular optimization has been explored with extensive applications in various domains, such as sensor placement [30], viral marketing [31], and robotic search problems [32]. To apply submodular optimization in realistic environments, different constraints (e.g., cardinality [24], additive budget [7], and routing [9]) must be considered.

Zhang et al. [9] propose a generalized cost-benefit (GCB) greedy algorithm to solve two NP-hard problems: monotone submodular maximization and routing path minimization (TSP). GCB finds solutions that achieves $\frac{1}{2}(1-\frac{1}{e})\widetilde{OPT}$ optimum, where $\widetilde{OPT}$ is the approximated optimum. However, there is a gap between $\widetilde{OPT}$ and the optimal solution (OPT). To improve the theoretical guarantee of GCB, Lin et al. [33] propose Tree-Structured Fourier Supports Set (TS-FSS) algorithms that combine the characteristics of submodularity and sparsity of routing trees to boost the theoretical bound.

Besides, Zhang et al. [34] investigate the non-monotone submodular maximization with the k-independence system routing constraint, where k is loosely upper-bounded by the size of the ground set. An iterated two-stage algorithm is presented to obatin a $[\frac{1}{4k}, 1+\theta]$-bicriterion approximation solution, where θ is an error parameter within $(0, 1)$.



(a) $f(S_A)$ (b) $f(S_B)$

(c) $f(S_A \cup s) - f(S_A) \geq f(S_B \cup s) - f(S_B)$

Figure 2.1: Illustration of submodularity under coverage scenario. The geographical locations of sensors are denoted as decimal numbers, while the blue and gray areas correspond to the covered and uncovered regions, respectively. (a) $f(S_A)$ is the covered area of S_A, where $S_A = \{1\}$. (b) $f(S_B)$ is the covered area of S_B, where $S_B = \{1, 2\}$. (c) The marginal gain of the coverage function f is marked as red dashed lines when $s = \{3\}$ is added to the current set $S_A = \{1\}$ and $S_B = \{1, 2\}$.

3 Background Knowledge

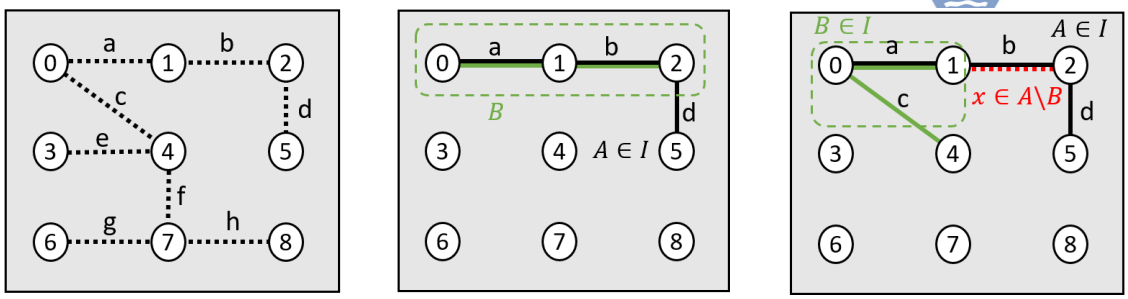
To formulate a multi-robot search problem, some preliminaries are introduced. Submodularity (Def. 1) and independence system (Def. 2) are adopted to formulate the objective function and the robot constraints, respectively. The objective function consists of the coverage and balancing function (Thm. 3.5). The constraint includes clustering of the search environment (Thm. 3.4). The advantage of formulating the submodular maximization subject to the intersection of an independence system (Thm. 3.1) is that the theoretical guarantee is given (Thm. 3.3). To further improve the performance, Def. 4 introduces the concept of matroid.

3.1 Submodularity

The definition and illustration of submodularity are as follows:

Definition 1. *(Submodularity) [24] A function* $f : 2^N \to \mathbb{R}^+$ *is submodular if and only if* $\forall S \subseteq T \subseteq N, \forall e \in N \setminus T,\ f(S \cup \{e\}) - f(S) \geq f(T \cup \{e\}) - f(S)$.

Let $S = \{1, 2, 3\}$ be the ground set and $S_A = \{1\}$, $S_B = \{1, 2\}$ be two different sets selected from the ground set, i.e., $S_A \subseteq S_B \subseteq S$. In Fig. 2.1 (a)(b), if the submodular function f is defined as the coverage function, $f(S_A)$ and $f(S_B)$ depict the regions observed by the camera FOV at locations of $\{1\}$ and $\{1, 2\}$. When a new set $s = \{2\}$ is added to both S_A and S_B, the marginal gain of the coverage function f is the additional regions covered, which is the red dashed line area shown in Fig. 2.1(c). The diminishing return property suggests that incorporating a set into the smaller set results in a greater marginal gain.



(a) Ground set E (b) $A \in \mathcal{I}$ and $B \subseteq A$ (c) $|A| > |B|$

Figure 3.1: Illustration of the downward closure and the exchange property. The vertices represent the geographic locations of sensors, and the edges are the distance between two vertices. (a) The ground set $E = \{a, b, c, d, e, f, g, h\}$ (the black dashed lines) (b) $A = \{a, b, d\}$ (the black solid lines) is a set that satisfies the independent set $\mathcal{I}$. $B = \{a, b\}$ (the green solid lines) is a subset of A. (c) Two sets $A = \{a, b, d\}$ (the black solid lines) and $B = \{a, c\}$ (the green solid lines) satisfy the independent set $\mathcal{I}$, where $|A| > |B|$. Then there exists an element $x \in A \setminus B$ such that adding to the smaller set B still satisfies the condition of the independent set $\mathcal{I}$, e.g., $x = b$ (the red dashed lines).

3.2 Matroid

Definition 2. *(Independence System) [35] An independence system is a pair $(E, \mathcal{I})$, where E is a finite set called the ground set and $\mathcal{I}$ is a family of subsets of E called independent sets such that:*

1. *(non-emptiness) $\emptyset \in \mathcal{I}$;*
2. *(downward closure) if $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$.*

An illustrative example is as follows. Consider a ground set $E = \{a, b, c, d, e, f, g, h\}$ and an independent set $\mathcal{I} = \{\{a, b, c\}, \{a, b, d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{a\}, \{b\}, \{c\}, \{d\}, \emptyset\}$, shown in Fig. 3.1(a). Let $A \in \mathcal{I}$ be a set, and $B \subseteq A$ be a subset. In Fig. 3.1(b), the downward closure property states that if $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$.

Definition 3. *(k-independence System) [35] Given a ground set E, an independence family $\mathcal{I}$ and a set $Y \subset E$. Let $r(Y)$ be the set of maximal elements of $\mathcal{I}$ which are subsets of Y. That is, $r(Y) = \{A \in \mathcal{I} | A \subseteq Y$ and there is no $A' \in \mathcal{I}$ such that $A \subset A' \subseteq Y\}$. Then $(E, \mathcal{I})$ is a k-independence system if for all $Y \subset E$,*

$$\frac{\max_{A \in r(Y)} |A|}{\min_{A \in r(Y)} |A|} \leq k,$$

where $k \geq 1$. As a special case, if $k = 1$, then $(E, \mathcal{I})$ is a matroid.

Definition 4. *(Matroid) [24] A matroid $\mathcal{M}$ is a pair $(E, \mathcal{I})$, where E is a finite set called the ground set and $\mathcal{I}$ is a family of subsets of E called independent sets, with the following properties:*

1. *(downward closure) if $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$;*
2. *(exchange property) if $A \in \mathcal{I}, B \in \mathcal{I}$ and $|A| > |B|$, then there exists $x \in A \setminus B$ such that $B \cup \{x\} \in \mathcal{I}$.*

An example of a matroid $\mathcal{M} = (E, \mathcal{I})$ is illustrated in Fig. 3.1. Consider a ground set $E = \{a, b, c, d, e, f, g, h\}$ and an independent set $\mathcal{I} = \{\{a, b, c\}, \{a, b, d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{a\}, \{b\}, \{c\}, \{d\}, \emptyset\}$ in Fig. 3.1(a). Let $A \in \mathcal{I}$ be a set and $B \subseteq A$ be a subset, shown in Fig. 3.1(b). The downward closure property states that if $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$. In Fig. 3.1(c), consider two sets $A \in \mathcal{I}$ and $B \in \mathcal{I}$, where $|A| > |B|$. By the exchange property, there exists an element $x \in A \setminus B$ such that $B + x \in \mathcal{I}$.

Theorem 3.1. *(Intersection of independence systems) [36] The intersection of a k_1-independence system and a k_2-independence system is a $(k_1 + k_2)$-independence system.*

To illustrate the concept, let $(E, \mathcal{I}_1)$ and $(E, \mathcal{I}_2)$ be a k_1-independence system and a k_2-independence system, respectively. The goal is to find the intersection of $(E, \mathcal{I}_1)$ and $(E, \mathcal{I}_2)$, namely $(E, \mathcal{I}_1 \cap \mathcal{I}_2)$. Then $(E, \mathcal{I}_1 \cap \mathcal{I}_2)$ is a $(k_1 + k_2)$-independence system.

Theorem 3.2. *(Intersection of matroids) [24] Consider the matroid intersection system*

$$\bigcap_{\mathcal{M} \in \mathbb{M}} \mathcal{M} \triangleq \left(\mathcal{V}, \bigcap_{\mathcal{M} \in \mathbb{M}} \mathcal{M} \right) \tag{3.1}$$

where $\mathbb{M}$ is a set of matroidal independence systems, $\mathcal{M}$ is a matroid, and $\mathcal{V}$ is a ground set. Eq. (3.1) models any arbitrary independence system.

Theorem 3.3. *(Theoretical bound of a matroid intersection) [37] Given a submodular monotone set function F and a set of matroidal independence systems $\mathbb{M}$, maximizing F under an intersection of matroids with the greedy algorithm yields a solution X. The performance of the set X is*

$$F(X) \geq \frac{1}{|\mathbb{M}| + 1} F(X^{opt}),$$

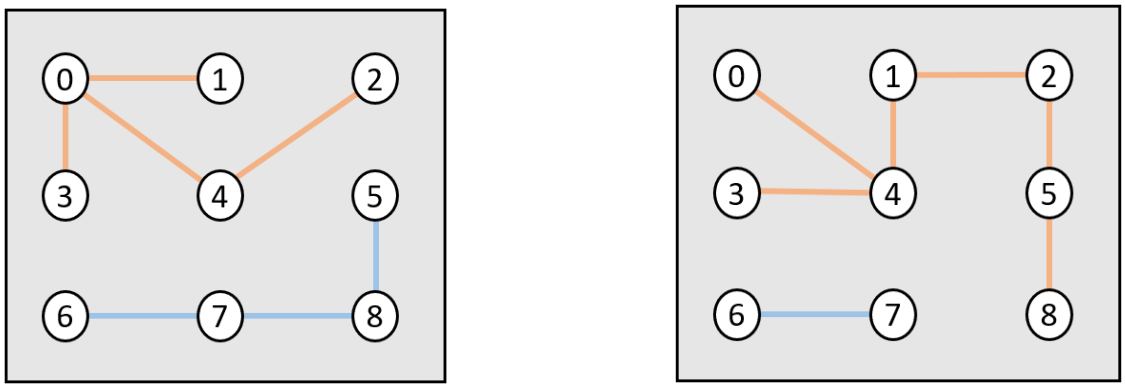
where $|\mathbb{M}|$ is the cardinality of $\mathcal{M}$ and X^{opt} is an optimal solution.

Consider a set of matroidal independence systems $\mathbb{M} = \{\mathcal{M}_1, \mathcal{M}_2\}$. The cardinality of $\mathbb{M}$ is 2. Therefore, greedy algorithms find a solution X such that the performance of X achieves $\frac{1}{3}F(X^{opt})$.

In multi-robot search scenarios, the search environment must be partitioned into smaller segments to accommodate multiple robots. Ensuring an equitable distribution of workloads among robots is a key consideration for efficient task assignments. Therefore, the coverage and balancing function [1] and the clustering constraint [38] are introduced.

Theorem 3.4. *(Clustering constraint) [38] Let E be the edge set and $S \in E$ be the set of subsets. $\mathcal{M}_C = (E, \mathcal{I}_C)$ is matroidal, where $\mathcal{I}_C = \{S \subseteq E : N \geq n\}$, N is the number of clusters, and n is the clustering budget.*

Theorem 3.5. *(Submodularity of a coverage and balancing function) [1] Let S be a set, $\lambda \in [0, 1]$ be a constant, $f : 2^S \to \mathbb{R}^+$ be a coverage function, and $\mathcal{B} : 2^S \to \mathbb{R}^+$ be a balancing function. The coverage and balancing function $F(S) = f(S) + \lambda\mathcal{B}(S)$ is submodular.*



(a) $\mathcal{B}(S_A) = -1.01$ (b) $\mathcal{B}(S_B) = -1.24$

Figure 3.2: Illustration of the balancing function. Two clusters are denoted by orange and blue colors. (a) $\mathcal{B}(S_A)$ is the balancing value of S_A, where $S_A = \{S_1, S_2\}$, $S_1 = \{0, 1, 2, 3, 4\}$ and $S_2 = \{5, 6, 7, 8\}$. (b) $\mathcal{B}(S_B)$ is the balancing value of S_B, where $S_B = \{S_1, S_2\}$, $S_1 = \{0, 1, 2, 3, 4, 5, 8\}$, and $S_2 = \{6, 7\}$.

Theorem 3.6. *(Submodularity of balancing function) [38] Given a graph* $\mathcal{G} = (V, E)$ *and a partition set* $S = \{S_i \in V : i \in \{1, ..., N\}\}$*, the balancing function* $\mathcal{B} : 2^E \to \mathbb{R}$ *is a monotonically increasing submodular function and is defined as follows:*

$$\mathcal{B}(S) = -\sum_i p_S(i) \log(p_S(i)) - N,$$

where $p_S(i) = \frac{|S_i|}{|V|}, i = \{1, ..., N\}$ *and* N *is the number of connected components in the graph.*

The balancing function encourages the clusters to have similar sizes. Fig. 3.2 shows the balancing function of two topologies. The balancing function shown in Fig. 3.2(a) has a higher value compared to that in Fig. 3.2(b). In other words, the higher balancing value has more balanced clustering.

4 Problem Formulation

Given a weighted graph $\mathcal{G} = (V, E, w)$ and an objective function $F : 2^E \to \mathbb{R}^+$, where V is a set of vertex, E is a set of edges and $w : E \to \mathbb{R}^+$ is a distance function, the goal is to find a subset $S \subseteq E$ that maximizes environmental coverage and balances robot workloads subject to a constraint. The objective function $F(S) = f(S)+\lambda\mathcal{B}(S)$ [1] is defined as a linear combination of a coverage function f and a balancing function $\mathcal{B}$, where $\lambda \in [0, 1]$ is the weight of the balancing function[1]. In 3D environments, the coverage function is defined by calculating the ratio of the number of covered voxels to the total number of environmental voxels.

Two variant formulations with known world maps are considered: Multi-robot search with independence system (MRSIS) [1] and Multi-robot search with matroid (MRSM).

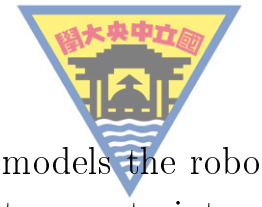
4.1 Multi-robot search with independence system (MRSIS)

Problem 1. *(MRSIS) [1] Given an objective function F, a set of independence system $\mathbb{I} = \{I_G, I_C\}$, the number of robots $n \in \mathbb{N}^+$, and a set of routing constraints for each robot $l = \{l_i \in \mathbb{R}^+\}_{i=1}^n$, find a subset $S \subseteq E$ that maximizes F under an intersection of independence systems*

$$\begin{aligned} \max_{S} \quad & F(S) \\ s.t. \quad & S \in \bigcap_{I \in \mathbb{I}} I, \end{aligned} \tag{4.1}$$

The intersection system $\bigcap_{I\in\mathbb{I}} I$ is defined as an intersection of all elements in the set of independence systems. $\mathbb{I} = \{I_G, I_C\}$ is defined as

[1] Both f and $\mathcal{B}$ functions are normalized.



a set of independence systems, where $I_G = (E, \mathcal{I}_G)$ models the robots' routing constraint and $I_C = (E, \mathcal{I}_C)$ models the cluster constraint.

The following theorems and corollaries prove that the routing and clustering constraints are k-independence systems. Then, the theoretical performance guarantee of MRSIS [1] is provided.

Theorem 4.1. *(Routing constraint of a k-independence system) The routing constraint for multiple robots $I_G = (E, \mathcal{I}_G)$ is a k_G-independence system, where $k_G > 1$. The independent set is defined as $\mathcal{I}_G = \{S \subseteq E : c(S \cap X_i) \leq l_i, \forall i \in \{1, ..., n\}\}$, where $c : 2^E \to \mathbb{R}^+$ is a routing cost function (e.g., TSP or MST solver), $X_i \subseteq S$ is the set of i^{th} cluster and l_i is a routing budget.*

Proof. To prove that I_G is an independence system, two properties described in Def. 2 must hold.

(Non-emptiness property) Consider a set $B_1 = \emptyset$. Let $A = \{1, ..., n\}$ and $m_i = c(B_1 \cap X_i), \forall i \in A$ be the routing costs of the i-th cluster and X_i be a set of cluster i. Since B_1 is an empty set, it implies that $m_i = 0, \forall i \in A$. Thus, B_1 belongs to the independent set $\mathcal{I}_G$.

(Downward closure property) Consider a set $B_1 \subseteq E$ and $B_1 \in \mathcal{I}_G$. Let $m_i = c(B_1 \cap X_i), \forall i \in A$ be the routing costs of the i-th cluster and X_i be a set of cluster i. Since $B_1 \in \mathcal{I}_G$, this implies that $c(B_1 \cap X_i) \leq l_i, \forall i \in A$. Therefore, $m_i \leq l_i, \forall i \in A$. Let $B_2 \subseteq B_1$ and $n_i = c(B_2 \cap X_i), \forall i \in A$. Since B_2 contains edges no more than B_1, the routing costs $n_i \leq m_i$. As a result, $n_i \leq m_i \leq l_i$. Therefore, B_2 belongs to the independent set $\mathcal{I}_G$.

Since the independent set $\mathcal{I}_G$ satisfies the non-emptiness and downward closure properties, $I_G = (E, \mathcal{I}_G)$ is a k_G-independence system. If I_G does not satisfy the exchange property (Def. 3.1), k_G will be greater than 1.

To prove that $k_G > 1$, consider two sets $B_1 \in \mathcal{I}$ and $B_2 \in \mathcal{I}$, where $|B_1| > |B_2|$. Let $s = B_1 \setminus B_2$, $m_i = c(B_1 \cap X_i) \leq l_i, \forall i \in A$ and $n_i = c(B_2 \cap X_i) = l_i, \forall i \in A$. If $X \notin \emptyset$, adding any element $x \in X$ to B_2 forms a new set B_2'. It is obvious that $B_2' \notin \mathcal{I}$ since $n_i = l_i \leq c(B_2' \cap X_i)$. Therefore, I_G does not satisfy the exchange property. k_G is greater than

1.

□

Corollary 4.1.1. *(Clustering constraint as a 1-independence system) Let E be an edge set and $S \subseteq E$ be a subset. $I_C = (E, \mathcal{I}_C)$ is an 1-independence system, where $\mathcal{I}_C = \{S \subseteq E : N \geq n\}$, N is the number of clusters, and n is the number of robots.*

Proof. Since I_C is a matroid (Thm. 3.4) and matroid satisfies the properties of the independence system (Def. 2), the clustering constraint I_C is an 1-independence system.

□

Theorem 4.2. *(Lower bound of MRSIS [1]) Let F be the coverage and balancing function, $\mathbb{I}$ be a set of k-independence systems, S be the solution of maximizing F subject to the intersection of $\mathbb{I}$ via greedy algorithms. The performance of S is*

$$F(X) \geq \frac{1}{2 + k_G} F(\overline{X^{opt}}),$$

where $k_G > 1$ and $\overline{X^{opt}}$ is an approximately optimal solution depending on the TSP or MST solver.

Proof. In Thm. 3.5, the coverage and balancing function F satisfies the submodularity. Let $\mathbb{I} = \{I_C, I_G\}$ be a set of k-independence systems. In Cor. 4.1.1, the clustering constraint I_C is a 1-independence system. In Thm. 4.1, the general multi-robot routing constraint I_G is a k_G-independence system. By Thm. 3.1, the performance of MRSIS [1] achieves $\frac{1}{2+k_G}\overline{OPT}$, where $\overline{OPT}$ is an approximately optimal solution that depends on the routing cost function.

□

To improve the performance of MRSIS [1], the set of independence systems is reformulated into the set of matroids.

4.2 Multi-robot search with matroid (MRSM)

Problem 2. *(MRSM) Given an objective function F, a set of matroidal independence systems $\mathbb{M} = \{\mathcal{M}_R, \mathcal{M}_C\}$, the number of robots $n \in \mathbb{N}^+$,*

and a set of routing constraints for each robot $l = \{l_i \in \mathbb{R}^+\}_{i=1}^n$, *find a set* $S \subseteq E$ *that maximizes* F *under an intersection of matroidal independence systems*

$$\begin{aligned} \max_{S} \quad & F(S) \\ s.t. \quad & S \in \bigcap_{\mathcal{M} \in \mathbb{M}} \mathcal{M}, \end{aligned} \tag{4.2}$$

$\mathbb{M} = \{\mathcal{M}_R, \mathcal{M}_C\}$ is defined as a set of matroidal independence systems, where $\mathcal{M}_R = (E, \mathcal{I}_R)$ and $\mathcal{M}_C = (E, \mathcal{I}_C)$ are the robots routing constraint under tree structure and the clustering constraint, respectively.

The routing matroid $\mathcal{M}_R$ constrains the trajectory length of robots while constructing spanning trees. The independent set $\mathcal{I}_R$ is a subset of E which satisfies:

1. $T \subset S$, where $T \subseteq E$ and $S \subseteq E$ are acyclic, and $|T| < |S|$,
2. $c_{st}(S \cap X_i) \leq l_i, \forall i \in \{1, 2, ..., n\}$.

where $X_i \subseteq S$ is the set of the i^{th} cluster and $c_{st} : 2^E \to \mathbb{R}^+$ is a routing cost function for a spanning tree.

The clustering matroid $\mathcal{M}_C$ is defined the same as in MRSIS [1]. The independent set $\mathcal{I}_C$ is defined as

$$\mathcal{I}_C = \{S \subseteq E : N \geq n\},$$

where N is the number of clusters and n is the number of robots.

The following Theorems introduce matroid constraints and the theoretical performance guarantee of MRSM.

Theorem 4.3. *(Routing constraint under tree structure) Let* E *be an edge set and* T *and* S *be any subset of* E. $\mathcal{I}_R$ *is the set of subsets,* $S \subseteq E$, *which satisfies:*

1. $T \subset S$, *where* T *and* S *are acyclic and* $|T| < |S|$,
2. $c_{st}(S \cap X_i) \leq l_i, \forall i \in \{1, ..., n\}$, *where* $c_{st} : 2^E \to \mathbb{R}^+$ *is a routing cost function for a spanning tree,* $X_i \subseteq S$ *is the set of* i^{th} *cluster and* l_i *is a routing budget.*

The routing constraint $\mathcal{M}_R = (E, \mathcal{I}_R)$ is a matroid.

Proof. To prove that $\mathcal{M}_R$ is a matroid, two properties described in Def. 4 must hold.

(Downward closure property) Consider a set $B_1 \subseteq E$ and $B_1 \in \mathcal{I}_R$. Let $A = \{1, ..., n\}$ and $m_i = c_{st}(B_1 \cap X_i), \forall i \in A$ be the routing costs of the i-th cluster, and X_i be the set of the i-th cluster. Since $B_1 \in \mathcal{I}_R$, this implies that B_1 is a tree (acyclic graph) and $m_i \leq l_i, \forall i \in A$. Let $B_2 \subseteq B_1$ and $n_i = c_{st}(B_2 \cap X_i), \forall i \in A$. Since B_2 contains edges no more than B_1, the routing costs must be $n_i \leq m_i$. In addition, removing any edge from B_1 forms B_2 which does not form a cycle. As a result, $n_i \leq m_i \leq l_i$ and B_2 is acyclic. Therefore, $\mathcal{I}_R$ satisfies the downward closure property.

(Exchange property) Consider two sets $B_1 \in \mathcal{I}_R$ and $B_2 \in \mathcal{I}_R$, where $|B_1| > |B_2|$. Since B_1 and B_2 satisfy the condition of the independent set and $|B_1| > |B_2|$, it implies that $B_2 \subset B_1$. Let $m_i = c_{st}(B_1 \cap X_i)$ and $n_i = c_{st}(B_2 \cap X_i), \forall i \in A$. If $B_2 \subset B_1$, it is clear that $n_i \leq m_i \leq l_i$. Besides, removing any edge from B_1 forms B_2, which is acyclic. Therefore, $\mathcal{I}_R$ satisfies the exchange property.

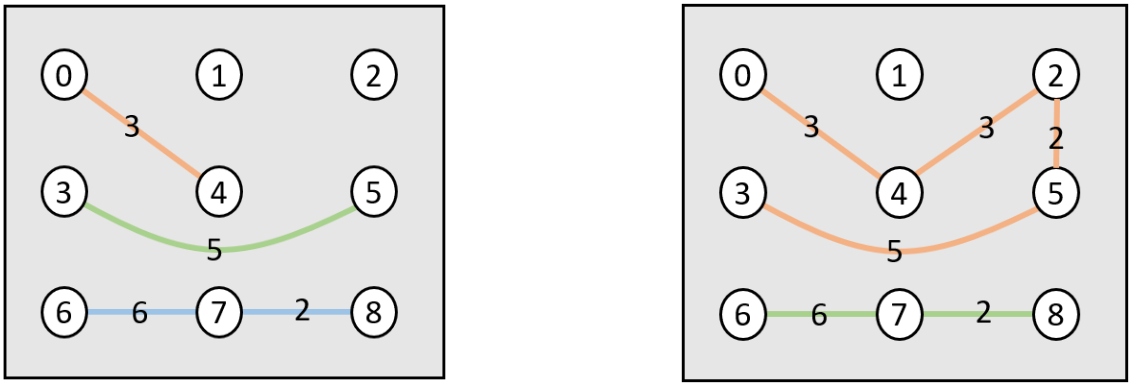
Since the independent set $\mathcal{I}_R$ satisfies the downward closure properties and the exchange property, $\mathcal{M}_R = (E, \mathcal{I}_R)$ is a matroid.

□

Definition 5. *(Matroidal independence systems of routing and clustering constraints) Given a tree-structured routing constraint $\mathcal{M}_R$ with a set of routing budgets $\{l_i\}_{i=1}^n$ and a clustering constraint $\mathcal{M}_C$ with n robots. A set of matroidal independence systems is defined as $\mathbb{M} = \{\mathcal{M}_R, \mathcal{M}_C\}$.*

The routing matroid $\mathcal{M}_R$ constrains the trajectory length of robots via constructing spanning trees while the clustering matroid $\mathcal{M}_C$ divides the ground set into groups for robots.

To illustrate the concept, two cases are illustrated as follows. Fig. 4.1 illustrates the intersection system of $\mathcal{M}_R = (E, \mathcal{I}_R)$ and $\mathcal{M}_C = (E, \mathcal{I}_C)$. Let E be the set of edges between any vertex, $(a, b) \in E$



(a) $S_A \in \bigcap_{\mathcal{M}\in\mathbb{M}} \mathcal{M}$ (b) $S_B \notin \bigcap_{\mathcal{M}\in\mathbb{M}} \mathcal{M}$

Figure 4.1: Illustration of the intersection system. The vertices and lines represent the subgoals and the distances between two vertices, respectively. Each color line denotes a cluster. (a) S_A contains three clusters that satisfy the condition of the intersection system. (b) S_B contains two clusters that violate the condition of the intersection system due to the under-budget of the clustering and the over-budget of routing constraints.

be an undirected edge, $\mathbb{M} = \{\mathcal{M}_R, \mathcal{M}_C\}$ be the set of matroidal independence systems, $l_i = 10$ be the routing constraint and $n = 3$ be the minimum number of clusters. In Fig. 4.1(a), let $S_A = \{S_1, S_2, S_3\}$, where $S_1 = \{(0,4)\}, S_2 = \{(3,5)\}$, and $S_3 = \{(6,7),(7,8)\}$. S_A satisfies the intersection system since S_A is acyclic, $|S_A| \geq n$ and $c(S_A \cap S_i) \leq l_i$, where $i = \{1,2,3\}$. In Fig. 4.1(b), let $S_B = \{S_1, S_2\}$, where $S_1 = \{(0,4),(4,2),(2,5),(3,5)\}$ and $S_2 = \{(6,7),(7,8)\}$. S_B does not satisfy the intersection system since $|S_B| < n$ and $c(S_B \cap S_1) > l_i$.

Theorem 4.4. *(Lower bound of the proposed algorithm for MRSM) Let F be the coverage and balancing function, $\mathbb{M}$ be a set of matroidal independence systems, S be the solution of maximizing F subject to the intersection of $\mathbb{M}$ via greedy algorithms. The performance of S is*

$$F(X) \geq \frac{1}{3} F(\widetilde{X^{opt}}),$$

where $\widetilde{X^{opt}}$ is the optimal solution under tree structure.

Proof. The coverage and balancing function F is submodular (Thm. 3.5) and each element in $\mathbb{M}$ is a matroid (Thm. 3.4 and Thm. 4.3). By Thm. 3.3, the performance of the proposed method is guaranteed to find a solution that achieves $\frac{1}{3}\widetilde{OPT}$, where $\widetilde{OPT}$ is the optimal performance under tree structure. □

5 Proposed Algorithm

This research proposes a Multi-robot Search with Matroid (MRSM) algorithm, which is an improved version of MRSIS [1]. The objective is to solve Eq. (4.2). The algorithm generates k trajectories for k robots.

MRSM is presented in Alg. 1. Line 1 is to initialize the selected edge set S. Lines 2-3 define the objective function and the set of matroidal independence systems, respectively. Line 5 is the edge initialization. Inspired by Kruskal's algorithm, the initial edge e^i is sampled based on the minimum edge weight. The main difference between MRSIS [1] and MRSM is lines 11-14. In MRSIS [1], the graph $S \cup \{e^*\}$ is transformed into robot trajectories by TSP or MST solver. In MRSM, the maximal marginal gain is selected to ensure that S remains spanning trees. The advantage is that MRSM rules out redundant processes (e.g., solving TSP or constructing MST) which improves the computational time. The computational complexity of MRSM is $\mathcal{O}(|E|^2)$. On the other hand, the computational complexity of MRSIS is $\mathcal{O}(|E|^2 + |E||V|^2)$ for both TSP and MST.

The key innovation of the MRSM algorithm is that it expands spanning trees with matroid property. Hence, MRSM adopts Kruskal's algorithm since Kruskal's algorithm inherently follows a matroid structure. In constructing an MST, Kruskal's algorithm selects the edge with the minimum weight from the ground set. It verifies the adding edge to the solution without generating cycling graphs. The primary difference between MRSM and Kruskal's algorithm is their objectives. Kruskal's algorithm seeks to generate a minimum spanning tree while MRSM discovers a spanning tree that maximizes the objective function under clustering and routing budgets.

Algorithm 1: Multi-robot Search with Matroid (MRSM)

Input: $G = (V, E, w)$ (weighted graph), $\mathbb{M}$ (set of matroidal independence systems), f (coverage function), $\mathcal{B}$ (balancing function), λ (balancing function weight), n (number of robots), l_i (routing budget)

Output: Selected edge set $S \subseteq E$

$S \leftarrow \emptyset$
$F \triangleq f + \lambda B$
$\mathbb{M} \triangleq \{\mathcal{M}_C, \mathcal{M}_R\}$
for $i \leftarrow 1$ to n **do**
  $e_i \leftarrow \text{SampleMin}(E, S)$
  $S \leftarrow S \cup \{e_i\}$
  $E \leftarrow E \setminus \{e_i\}$
end for
while $E \neq \emptyset$ **do**
  $e^* \leftarrow \arg\max_{e \in E} F(S \cup e) - F(S)$
  $S' \leftarrow S \cup \{e^*\}$
  if $S' \in \bigcap_{\mathcal{M} \in \mathbb{M}} \mathcal{M}$ **then**
    $S \leftarrow S \cup \{e^*\}$
  end if
  $E \leftarrow E \setminus \{e^*\}$
end while

6 Experiments

The proposed algorithm (MRSM) and the benchmark algorithms (MRSIS [1], CapAM [2], PD-FAC [3], and the other 2 methods) are evaluated based on the expected number of detected objects (ENDO) and the expected time to detection (ETTD). The ETTD is defined as follows:

$$E[TTD] = \frac{1}{Nn} \sum_{j=1}^{N} \sum_{i=1}^{n} t_i^j,$$

where N, n, and t_i^j represent the number of trials, number of objects in trial $j \in N$, and detection time of object $i \in n$ in trial $j \in N$. Besides, if the drones fail to detect objects, the search time is set to the maximal time constraint.

The search process is as follows: MRSM generates a set of trajectories composed of K_i edges before the i^{th} drone takes off. As the drone reaches the subgoal, it hovers and detects targets within 3 seconds. The drone continues to traverse the edge and visits the next subgoal. The process continues until all targets in the environment have been identified or all drones have traversed their designated edges. When the search task is terminated, drones land on the ground.

6.1 Experiment Setup

Three experiments are conducted: the target search experiment (EX1), the parametric analysis (EX2), and the scalability analysis (EX3).

In the target search experiment (EX1), the performance of the proposed algorithms (MRSM) is assessed through simulation and a realistic environment (see Fig. 6.2). The assessment includes a comparison with state-of-the-art approaches (e.g., MRSIS [1], CapAM [2] and PD-FAC [3]). The simulation is carried out with 3 robots (UAVs) on Gazebo (see Fig. 6.1). Various environment sizes $E = \{8, 12\}$ and numbers of

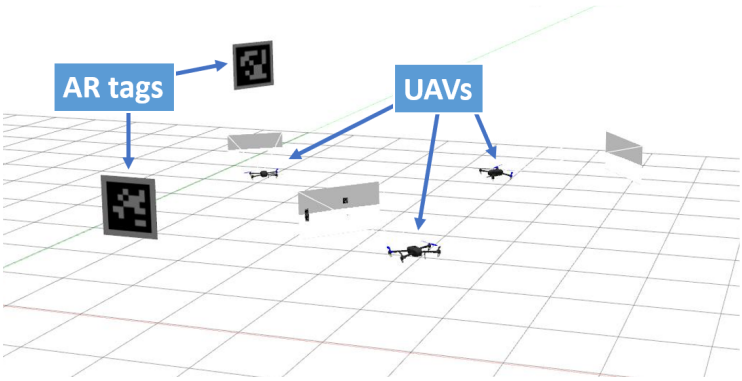


Figure 6.1: Gazebo simulation with 3 UAVs and 2 objects (AR Tags) in a 3D Environment. The UAV can project a viewing frustum to observe the search space.

targets (AR tags) $T = \{2, 4, 6\}$ in the space are evaluated. Each (e, t) pair generates 100 trials with randomly located targets, where $e \in E$ is an $e \times e \times e(m.)$ search space, and $t \in T$ is the number of targets in e. The robot's goal is to find all targets within scenarios of partial occlusion.

A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ is constructed to represent the environment, where $\mathcal{V}$ is the subgoal set, $\mathcal{E}$ is the edge set, and c is the routing costs with respect to edges. The subgoals $\mathcal{V}$ are evenly distributed throughout the space and each subgoal is defined as $v \in \{(x, y, z, \theta)|(x, y, z, \theta) \in \mathcal{V}\}$, where $x, y, z \in [1, e]$ and $\theta \in \{0, 60, 120, 180, 240, 300\}$.

In the search process, each robot can take one of the actions from $\mathcal{A} = \{Move(x, y, z, \theta), Detect\}$. The $Move$ action takes the drone to the coordinate (x, y, z) with the heading θ. The $Detect$ action performs an object detection. The time constraint of the search process is 10 minutes. Once all the targets have been found or the search time exceeds the constraint, the task is terminated.

The coverage function (f) is calculated by the ratio of the number of covered voxels to the total number of environmental voxels. The routing cost (c) is measured by the Euclidean distance between two subgoals[1].

The real-world experiment is conducted with 2 drones in a $13 \times 9\ m$ public area on the third floor of the General Education Building at the

[1] For any two connected nodes, $v_1 = (x_1, y_1, z_1, \theta_1)$ and $v_2 = (x_2, y_2, z_2, \theta_2)$, the routing cost $c(\{(v_1, v_2)\}) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 + (\theta_1 - \theta_2)^2}$.

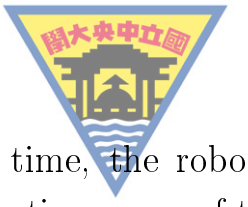
National Central University. The map and subgoals are shown in Fig. 6.2(a). The space is divided into voxels whose unit size is $15 \times 15 \times 15\ cm$. The goal is to find a sports ball, a chair, and a bottle in the environment, shown in Fig. 6.2(b). Three targets are randomly located and can be partially occluded due to the complex environment with obstacles. The searchers are the drones developed by Taiwan Drone 100 shown in Fig. 6.3. The drone is equipped with NVIDIA Jetson Xavier NX and two Intel RealSense cameras. The Intel RealSense T265 camera is used to localize the drone and the D435i camera is to explore the environment. The camera and drone parameters are shown in Table 6.1.

To detect objects, the YOLOv5 [39] is adopted and run on NVIDIA Jetson Xavier NX. The uncertainty of detection is considered due to object occlusion in the environment. To successfully detect the object, the confidence rate of detection must be over a threshold of 0.6.

In the parametric analysis experiment (EX2), the parameters, the weight of the balancing function (λ), and the robot routing budget (l_i) are investigated for MRSIS [1] and MRSM algorithms within an $8 \times 8 \times 8(m.)$ simulated environment. Three robots (UAVs) are tasked to find various numbers of targets $t \in \{2, 4, 6\}$ with different balancing weights $\lambda \in \{0\,, 0.2,\ 0.4,\ 0.6,\ 0.8,\ 1\}$ and robot routing budgets $l_i \in \{20,\ 40,\ 60,\ 80\}$. When $\lambda = 0$, no balanced workloads are considered. When $\lambda = 1$, the workloads assigned to robots are thoroughly optimized. The experiment setup is the same as the simulation in EX1. Additionally, a comparative analysis of computational time is conducted for both methods.

In the scalability analysis experiment (EX3), the performance of MRSM and MRSIS [1] is evaluated under a $20 \times 20 \times 20(m^3)$ simulated space involving a considerable number of robots $R = \{5, 10\}$. Robots (UAVs) must find 8 targets (AR tags) in the environment. The balancing weight λ and robot routing budget l_i are set to 0.2 and 300, respectively. The camera range is set to 5 meters and the remaining configurations are established in the same manner as the simulation in EX1.

Two baseline methods (CapAM [2] and PD-FAC [3]) are implemented for the search scenarios. In CapAM [2], the state space of the



MDP contains features such as the elapsed mission time, the robot's current location, and the robot's work capacity. The action space of the MDP is the action set $\mathcal{A}$. The goal of CapAM [2] is to visit as many subgoals as possible. In PD-FAC [3], the weighted graph $\mathcal{G}$ maintains the same format as detailed in [3]. The search time budget is set to 10 minutes and the search targets are randomly placed on the graph nodes. Once the robot visits the node on the graph, the target is marked as found. The configurations of PD-FAC can be found in [3].

Table 6.1: Parameters of EX1 and EX2.

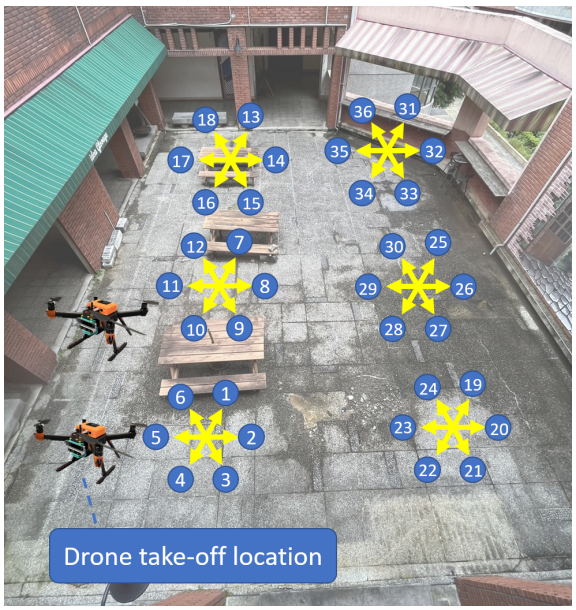
Parameters	Simulation	Real-world search
Camera range	$2m$	$3m$
Horizontal FOV	45°	69°
Vertical FOV	45°	42°
Transitional velocity	1.3 m/sec	0.2 m/sec
Angular velocity	45 deg/sec	120 deg/sec

6.2 EX1: Comparison with Benchmarks on Targets Search

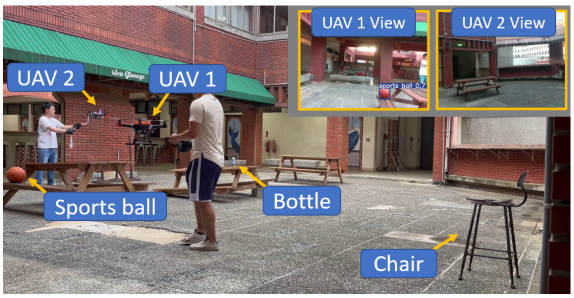
Table 6.2 and Table 6.3 present the simulated performance with a routing budget $l_i = 350$, the number of robots $k = 3$ and a balancing function weight $\lambda = 1$ across various environment sizes. MRSM is compared to state-of-the-art baselines (e.g., MRSIS [1], CapAM [2], AM-RL [40], and PD-FAC [3]).

In Table 6.2, drones find more objects in a small environment ($e = 8$) compared to the larger one ($e = 12$). The decrease in the number of targets in the environment correlates with an increase in the difficulty of the search. MRSM achieves the highest ENDO in the large environment ($e = 12$). The advantage of MRSM lies in its ability to maximize both the coverage function and the workload balancing function under matroid constraints. In addition, MRSIS-TSP outperforms MRSM in the case of $e = 8$. The detailed performance of both MRSM and MRSIS-TSP [1] will be examined in EX2.

Table 6.3 shows the ETTD of all methods. All approaches are



(a)



(b)

Figure 6.2: (a) A 13×9 m. public area on the third floor of the General Education Building at the National Central University. (b) An example of UAVs searching for targets (a sports ball, a bottle, and a chair).

Figure 6.3: A customized UAV developed by Taiwan Drone 100.

Table 6.2: The expected number of detected objects (ENDO) in various environment sizes (E) and number of targets (T) on Gazebo simulator.

Env. size (E)	8			12		
Num. of targets (T)	2	4	6	2	4	6
Random	0.86	1.43	2.25	0.12	0.34	0.52
AM-RL [40]	**1.5**	**3.15**	**4.73**	0.23	0.56	0.77
CapAM [2]	0.89	1.69	2.48	0.34	0.81	1.18
PD-FAC [3]	0.49	0.81	1.11	0.15	0.43	0.46
MRSIS-MST [1]	1.03	2.14	3.21	0.51	0.99	1.51
MRSIS-TSP [1]	1.46	2.92	4.41	1.60	3.01	4.88
MRSM	1.44	2.92	4.39	**1.68**	**3.19**	**4.97**

constrained to search within 10 minutes (600 seconds). For $e = 8$, MRSIS-MST [1] is the most efficient, while for $e = 12$, MRSM takes the lead in efficiency. MRSM, in comparison to AM-RL [40], detects fewer average targets in the $e = 8$ scenario. Despite AM-RL [40] finding the highest ENDO, especially within the time constraint of 600 seconds, its search efficiency is compromised due to the largest ETTD value.

To further examine the proposed approaches, algorithms (MRSIS [1], CapAM [2], and PD-FAC [3]) are evaluated in a real-world environment. Experiments are conducted 6 times for each approach. The ETTD and the coverage rate are shown in Table 6.4. MRSM achieves an ETTD of 201 seconds and a coverage rate of 67%, which is the lowest ETTD and the highest coverage rate among other approaches. Unlike the MST approach (MRSIS-MST [1]), MRSM generates spanning trees that may not be the shortest trajectories, but are the most efficient for target discovery. The experiments demonstrate that MRSM outperforms benchmark algorithms.

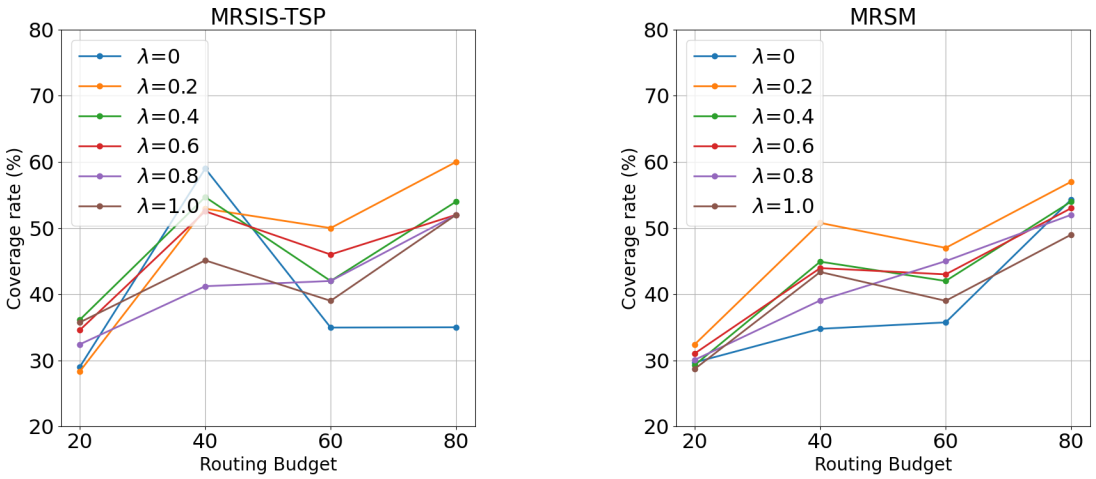
The summaries of these experiments are as follows. The proposed algorithm (MRSM) outperforms state-of-the-art approaches (e.g., MRSIS [1], AM-RL [40], CapAM [2], and PD-FAC [3]) as claimed in Thm. 4.4. MRSM achieves the highest coverage rate and discovers targets with the lowest ETTD. However, it is not guaranteed to find the most targets in the environment.

Table 6.3: The expected time to detection (ETTD) in various environment sizes (E) and number of targets (T) on Gazebo simulator. The unit is seconds.

Env. size (E)	8			12		
Num. of targets (T)	2	4	6	2	4	6
Random	480	438	428	600	600	600
AM-RL [40]	600	600	600	565	540	518
CapAM [2]	463	418	377	544	500	463
PD-FAC [3]	525	450	439	556	497	486
MRSIS-MST [1]	**251**	**211**	**205**	532	478	454
MRSIS-TSP [1]	264	273	259	379	416	386
MRSM	260	271	258	**365**	**392**	**368**

Table 6.4: The expected time to detection (ETTD) of MRSM and baselines (MRSIS [1], CapAM [2], and PD-FAC [3]) in the real-world environment.

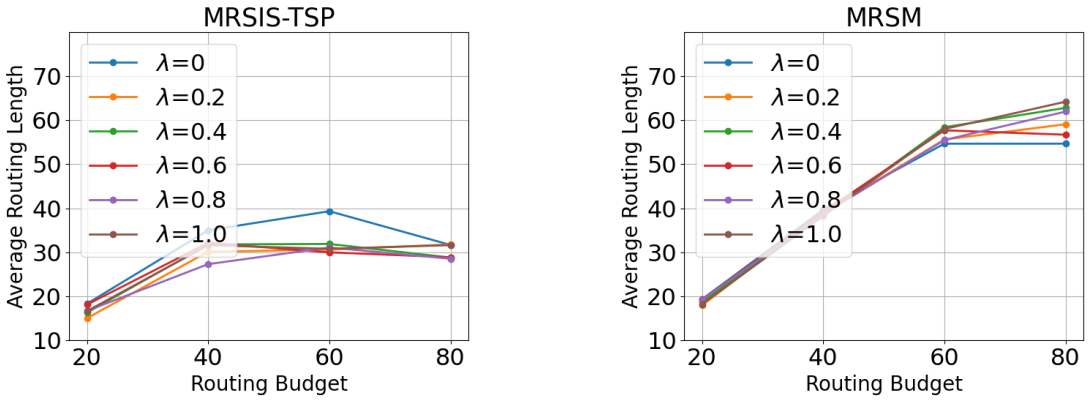
Method	Mean (sec.)	Std.	Coverage Rate
CapAM [2]	275	93	42%
PD-FAC [3]	314	130	33%
MRSIS-MST [1]	206	75	65%
MRSIS-TSP [1]	280	58	62%
MRSM	**201**	93	**67%**



(a) MRSIS-TSP [1]

(b) MRSM

Figure 6.4: Coverage rate with different robot routing budget l_i and balancing weight λ.



(a) MRSIS-TSP [1]

(b) MRSM

Figure 6.5: Average routing length (in meters) with various balancing weights λ and robot routing budget.

Table 6.5: The expected number of detected objects (ENDO) with different parameters (robot routing budget l_i, number of targets t, and balancing weight λ) on Gazebo simulator.

l_i	t	Method	λ					
			0	0.2	0.4	0.6	0.8	1
20	2	MRSIS-TSP [1]	0.8	1.0	1.1	1.0	1.1	1.1
		MRSM	**1.1**	**1.1**	**1.2**	**1.2**	**1.1**	**1.1**
	4	MRSIS-TSP [1]	1.8	2.1	2.3	2.3	2.2	2.0
		MRSM	**2.3**	**2.2**	**2.3**	**2.4**	**2.2**	**2.2**
	6	MRSIS-TSP [1]	2.6	3.2	3.4	3.4	3.2	3.1
		MRSM	**3.4**	**3.3**	**3.4**	**3.6**	**3.2**	**3.3**
40	2	MRSIS-TSP [1]	**1.5**	1.3	1.4	1.4	1.2	**1.5**
		MRSM	1.2	**1.4**	**1.4**	**1.4**	**1.3**	1.3
	4	MRSIS-TSP [1]	**3.2**	2.8	2.8	**3.1**	2.6	**3.0**
		MRSM	2.4	**2.9**	**2.8**	2.9	**2.7**	2.8
	6	MRSIS-TSP [1]	**4.6**	4.1	4.2	**4.5**	3.8	**4.5**
		MRSM	3.5	**4.5**	**4.3**	4.3	**4.1**	4.1
60	2	MRSIS-TSP [1]	**1.3**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
		MRSM	1.3	0.9	0.9	0.9	0.9	0.9
	4	MRSIS-TSP [1]	2.6	**2.2**	**2.0**	**2.2**	**2.2**	**2.1**
		MRSM	**2.7**	2.1	1.9	2.0	2.0	1.8
	6	MRSIS-TSP [1]	3.7	**3.3**	**3.0**	**3.2**	**3.3**	**3.2**
		MRSM	**4.0**	3.1	2.9	2.9	3	2.8
80	2	MRSIS-TSP [1]	1.3	**1.5**	**1.5**	1.4	**1.4**	**1.4**
		MRSM	**1.5**	1.4	1.5	**1.5**	1.4	1.4
	4	MRSIS-TSP [1]	2.7	**3.2**	**3.0**	2.8	2.9	**3.1**
		MRSM	**3.1**	3.1	3.0	**3.1**	**3.0**	2.9
	6	MRSIS-TSP [1]	4	**4.6**	**4.4**	4.1	4.3	**4.4**
		MRSM	**4.7**	4.4	4.4	**4.5**	**4.4**	4.3

Table 6.6: The expected time to detection (ETTD) with different parameters (robot routing budget l_i, number of targets t, and balancing weight λ) on Gazebo simulator. The unit is in seconds.

l_i	t	Method	λ					
			0	0.2	0.4	0.6	0.8	1
20	2	MRSIS-TSP [1]	384	342	335	347	328	342
		MRSM	**326**	**315**	**303**	**282**	**323**	**328**
	4	MRSIS-TSP [1]	382	340	**313**	325	337	355
		MRSM	**308**	**313**	315	**292**	**317**	**328**
	6	MRSIS-TSP [1]	378	334	**313**	316	330	348
		MRSM	**309**	**313**	314	**292**	**327**	**320**
40	2	MRSIS-TSP [1]	**272**	302	282	271	316	**258**
		MRSM	309	**261**	**278**	**261**	**293**	300
	4	MRSIS-TSP [1]	**251**	275	289	**249**	292	**261**
		MRSM	329	**264**	**286**	252	**279**	273
	6	MRSIS-TSP [1]	**260**	278	277	249	292	**247**
		MRSM	320	**246**	**255**	**244**	**277**	269
60	2	MRSIS-TSP [1]	**307**	**266**	**270**	**283**	**266**	**297**
		MRSM	308	317	318	307	324	328
	4	MRSIS-TSP [1]	**310**	**195**	**203**	**223**	**205**	**226**
		MRSM	312	229	271	243	238	249
	6	MRSIS-TSP [1]	321	**159**	**158**	**190**	**148**	**179**
		MRSM	**301**	205	231	194	209	204
80	2	MRSIS-TSP [1]	313	**277**	**273**	286	**275**	**263**
		MRSM	**269**	288	274	**273**	286	280
	4	MRSIS-TSP [1]	293	**256**	**275**	288	**266**	**242**
		MRSM	**286**	257	280	**267**	279	290
	6	MRSIS-TSP [1]	291	**260**	**274**	286	**256**	**249**
		MRSM	**271**	270	271	**262**	269	278

6.3 EX2: Parametric Analysis

To verify the parametric analysis of MRSIS-TSP [1] and MRSM, the parameters (the coverage rate, the average routing length, the ENDO, and the ETTD) are compared in various balancing weights (λ) and robot routing budgets (l_i).

The coverage rate of MRSIS-TSP [1] and MRSM under different balancing weights (λ) and robot routing budgets (l_i) is as follows. Fig. 6.4(a) illustrates the coverage rate of MRSIS-TSP [1]. When $\lambda = 0$, a substantial variance exists among the routing budgets. No balancing function outperforms at $l_i = 40$ while performing less effectively in other cases. Besides, a smaller balancing weight demonstrates effective performance. At $l_i = 20$, $l_i = 60$, and $l_i = 80$, a small magnitude of the balancing weight performs optimally. At $l_i = 40$, no balancing function ranks the highest. However, when the balancing function is considered, $\lambda = 0.4$ achieves the highest coverage rate. Fig. 6.4(b) illustrates the coverage rate of MRSM. The weight $\lambda = 0.2$ yields the highest coverage rate across all routing budget cases. Besides, the coverage rate of MRSM is more consistent than that of MRSIS-TSP [1].

Fig. 6.5 illustrates the average routing length on 3 robots with various balancing weights (λ) and robot routing budgets (l_i). As the routing budget increases, robots cover larger traveling distances. The routing length converges due to the limitation of environment size. In MRSIS-TSP [1] scenario (Fig. 6.5(a)), robots travel between 10 and 40 meters. In MRSM scenario (Fig. 6.5(b)), the traveled distance is between 20 and 65 meters. The reason for the smaller routing distance in MRSIS-TSP [1] compared to MRSM is that MRSIS-TSP [1] utilizes a TSP approximator to minimize travel distances. Therefore, the average routing length of MRSM is more than that of MRSIS-TSP [1].

Table 6.5 and Table 6.6 show the ENDO and the ETTD, respectively. The best performance is highlighted in bold. The results indicate that no single method consistently outperforms in all cases. On average, MRSM outperforms in 38 out of 72 cases, while MRSIS-TSP [1] achieves superiority in 34 out of 72 cases. Additionally, MRSM excels in small routing budgets ($l_i \leq 40$), while MRSIS-TSP [1] performs well in

the larger ones. Since MRSIS-TSP [1] inherits the routing minimization feature, it has an advantage in larger routing budgets by facilitating efficient travel. Moreover, in some cases, the performance is superior when the balancing function is not involved. Since the proposed objective function is defined as $F(S) = f(S) + \lambda\mathcal{B}(S)$, when $\lambda = 0$, the objective function is simply a coverage function. Optimizing the coverage function with routing and clustering constraints may yield good performance in certain cases.

Hence, the EX2 results demonstrate that a small balancing weight magnitude leads to optimal performance for both MRSM and MRSIS-TSP [1]. Across most cases, MRSM outperforms MRSIS-TSP [1] due to the theoretical guarantee of submodularity under matroid constraints.

6.4 EX3: Scalability Analysis

Table 6.7 shows the performance of MRSM, CapAM [2], and PD-FAC [3] with various number of robots (UAVs). If all subgoals are selected, the coverage is 79%.

The coverage rate increases with the number of robots. The results show that CapAM [2] identifies the highest number of targets and achieves the largest coverage in both scenarios (5 UAVs and 10 UAVs). Meanwhile, MRSM achieves the lowest ETTD, while the ENDO of MRSM closely approaches that of CapAM [2]. The difference between MRSM and CapAM [2] is that MRSM maximizes the coverage rate and the workload balance, while CapAM [2] maximizes the number of visited subgoals.

In summary, due to the theoretical guarantee of submodularity, when the number of robots increases, ETTD does not deteriorate. In addition, MRSM outperforms state-of-the-art methods for multi-robot search scenarios.

Table 6.7: Large-scale experiment results of the ENDO, the ETTD, and the coverage rate with different numbers of UAVs on Gazebo simulator.

No. UAVs	Method	ENDO	ETTD	Coverage rate
5	CapAM [2]	**6.8**	569	**79%**
	PD-FAC [3]	3.9	416	44%
	MRSM	6.2	**285**	71%
10	CapAM [2]	**6.8**	421	**79%**
	PD-FAC [3]	4.2	344	47%
	MRSM	6.4	**214**	74%

7 Conclusions and Future Work

To overcome the coverage and task allocation problems, this research proposes the MRSM algorithm. In addition, the clustering and routing constraints are reformulated into the intersection of matroids. With submodularity and matroid property, MRSM achieves $\frac{1}{3}\widetilde{OPT}$, where $\widetilde{OPT}$ is an approximately optimal performance under the spanning tree structure. Experiment results show that MRSM outperforms state-of-the-art approaches in multi-robot search problems.

The future work of this research is as follows. First, MRSM relies on known environments. The multi-robot search with an unknown map can be achieved by map exploration and a robot search approach. [41] Second, MRSM is an offline planning approach. Extending the current approach to plan a trajectory online via adaptive submodularity is another direction. [31] Third, in EX2, where MRSM does not consistently surpass MRSIS-TSP [1], exploring the integration of both advantages becomes another viable direction.

References

[1] Yan-Shuo Li and Kuo-Shih Tseng. Multi-robot search in a 3d environment with intersection system constraints. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[2] Steve Paull, Payam Ghassemi, and Souma Chowdhury. Learning scalable policies over graphs for multi-robot task allocation using capsule attention networks. *International Conference on Robotics and Automation (ICRA)*, pages 8815–8822, 2022.

[3] Wenda Sheng, Hongliang Guo, Wei-Yun Yau, and Yingjie Zhou. Pd-fac: Probability density factorized multi-agent distributional reinforcement learning for multi-robot reliable search. *IEEE Robotics and Automation Letters*, 7(4):8869–8876, 2022.

[4] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032, 2018.

[5] James S Jennings, Greg Whelan, and William F Evans. Cooperative search and rescue with a team of mobile robots. *International Conference on Advanced Robotics (ICAR)*, pages 193–200, 1997.

[6] Nikolaus Correll and Alcherio Martinoli. Multirobot inspection of industrial machinery. *IEEE Robotics & Automation Magazine*, 16(1):103–112, 2009.

[7] Samir Khuller, Anna Moss, and Joseph Seffi Naor. The budgeted maximum coverage problem. *Information processing letters*, 70(1):39–45, 1999.

[8] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[9] Haifeng Zhang and Yevgeniy Vorobeychik. Submodular optimization with routing constraints. *Association for the Advancement of Artificial Intelligence (AAAI)*, 30(1), 2016.

[10] Amarjeet Singh, Andreas Krause, Carlos Guestrin, William Kaiser, and Maxim Batalin. Efficient planning of informative paths for multiple robots. *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI)*, pages 2204–2211, 2007.

[11] Yan-Shuo Li and Kuo-Shih Tseng. Computation-aware multi-object search in 3d space using submodular tree. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[12] Yan-Shuo Li and Kuo-Shih Tseng. Multi-robot search in 3d environments using submodularity with matroid intersection constraints. *Submitted to IEEE Transactions on Robotics*, 2024.

[13] Boyu Zhou, Yichen Zhang, Xinyi Chen, and Shaojie Shen. Fuel: Fast uav exploration using incremental frontier structure and hierarchical planning. *IEEE Robotics and Automation Letters*, 6(2):779–786, 2021.

[14] Boyu Zhou, Hao Xu, and Shaojie Shen. Racer: Rapid collaborative exploration with a decentralized multi-uav system. *IEEE Transactions on Robotics*, 2023.

[15] Mayank Mittal, Rohit Mohan, Wolfram Burgard, and Abhinav Valada. Vision-based autonomous uav navigation and landing for urban search and rescue. *The International Symposium of Robotics Research*, pages 575–592, 2019.

[16] Mikko Lauri, Joni Pajarinen, Jan Peters, and Simone Frintrop. Multi-sensor next-best-view planning as matroid-constrained submodular maximization. *IEEE Robotics and Automation Letters*, 5(4):5323–5330, 2020.

[17] Sharaf C. Mohamed, Sanjif Rajaratnam, Seung Tae Hong, and Goldie Nejat. Person finding: An autonomous robot search method for finding multiple dynamic users in human-centered environments. *IEEE Transactions on Automation Science and Engineering*, 17(1):433–449, 2020.

[18] Sharaf C. Mohamed, Angus Fung, and Goldie Nejat. A multirobot person search system for finding multiple dynamic users in human-centered environments. *IEEE Transactions on Cybernetics*, pages 1–13, 2022.

[19] Xiaolong Zhu, Fernando Vanegas, and Felipe Gonzalez. An approach for multi-uav system navigation and target finding in cluttered environments. *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1113–1120, 2020.

[20] Kaiyu Zheng, Yoonchang Sung, George Konidaris, and Stefanie Tellex. Multi-resolution pomdp planning for multi-object search in 3d. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2022–2029, 2021.

[21] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.

[22] Jun Liu and Ryan K Williams. Optimal intermittent deployment and sensor selection for environmental sensing with multi-robot teams. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1078–1083, 2018.

[23] Ted K Ralphs, Leonid Kopman, William R Pulleyblank, and Leslie E Trotter. On the capacitated vehicle routing problem. *Mathematical programming*, 94:343–359, 2003.

[24] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[25] Stefan Jorgensen, Robert H Chen, Mark B Milam, and Marco Pavone. The matroid team surviving orienteers problem: Constrained routing of heterogeneous teams with risky traversal. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5622–5629, 2017.

[26] Jun Liu and Ryan K Williams. Submodular optimization for coupled task allocation and intermittent deployment problems. *IEEE Robotics and Automation Letters*, 4(4):3169–3176, 2019.

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

[28] Ekaterina Tolstaya, James Paulos, Vijay Kumar, and Alejandro Ribeiro. Multi-robot coverage and exploration using spatial graph neural networks. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8944–8950, 2021.

[29] Hao Zhang, Jiyu Cheng, Lin Zhang, Yibin Li, and Wei Zhang. H2gnn: hierarchical-hops graph neural networks for multi-robot exploration in unknown environments. *IEEE Robotics and Automation Letters*, 7(2):3435–3442, 2022.

[30] Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. *Association for the Advancement of Artificial Intelligence (AAAI)*, 7:1650–1654, 2007.

[31] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[32] Yu-Chung Tsai, Bing-Xian Lu, and Kuo-Shih Tseng. Spatial search via adaptive submodularity and deep learning. *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 112–113, 2019.

[33] Pao-Te Lin and Kuo-Shih Tseng. Improvement of submodular maximization problems with routing constraints via submodularity and

fourier sparsity. *IEEE Robotics and Automation Letters*, pages 1–8, 2023.

[34] Haotian Zhang, Rao Li, Zewei Wu, and Guodong Sun. Nonmonotone submodular maximization under routing constraints. *arXiv preprint arXiv:2211.17131*, 2022.

[35] Bernhard Korte and Dirk Hausmann. An analysis of the greedy heuristic for independence systems. 2:65–74, 1978.

[36] Julián Mestre. On the intersection of independence systems. *Operations Research Letters*, 43(1):7–9, 2015.

[37] Marshall L Fisher, George L Nemhauser, and Laurence A Wolsey. *An analysis of approximations for maximizing submodular set functions—II*. Springer, 1978.

[38] Ming-Yu Liu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. Entropy-rate clustering: Cluster analysis via maximizing a submodular function subject to a matroid constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):99–112, 2013.

[39] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, November 2022.

[40] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! *International Conference on Learning Representations*, 2019.

[41] Bing-Xian Lu and Kuo-Shih Tseng. 3d map exploration via learning submodular functions in the fourier domain. *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1199–1205, 2020.