2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Computation-Aware Multi-object Search in 3D Space using Submodular Tree

Yan-Shuo Li[1] and Kuo-Shih Tseng[2]

***Abstract*— Searching for targets in 3D environments can be formulated as submodular maximization problems with routing constraints. However, it involves solving two NP-hard problems: the maximal coverage problem and the traveling salesman problem. Since the time constraint is critical for search problems, this research proposes a Computation-Aware Search for Multiple Objects (CASMO) algorithm to further consider the computational time in the cost constraints. Due to the submdularity, the greedy algorithm achieves $\frac{1}{2}(1-\frac{1}{e})\overline{OPT}$, where $\overline{OPT}$ is the approximate optimum. The experiment results show that the proposed algorithm outperforms state-of-the-art approaches in multi-object search.**

I. Introduction

Efficient search in 3D environments is a key technology in robotics for various applications (e.g., environmental monitoring [1], infrastructure inspection [2], and industrial automation [3]). Search problems can be reformulated as a submodular maximization problem with routing constraints [4]. However, it involves two NP-hard problems: the maximal coverage problem and the traveling salesman problem. The maximal coverage problem is to find the maximal coverage of K locations from N ground sets. The traveling salesman problem is to find the least-cost route from K locations within the routing constraint. Thanks to submodularity, greedy approaches provide solutions with theoretical guarantees [5] [6].

State-of-the-art approaches formulate robotic search problems as Partially Observable Markov Decision Processes (POMDPs). In [7], the researchers propose a multi-resolution planning algorithm based on online Monte Carlo tree search (MCTS) methods. However, its computation is inefficient, as it expands MCTS trees during the search for targets. Furthermore, when the target locations change in the environment, the search strategy requires to be re-planned.

To enhance computational efficiency, this research proposes a Computation-Aware Search for Multiple Objects (CASMO) algorithm based on the Tree-Structured Fourier Supports Set (TS-FSS) algorithm [5]. The key difference is that the computational cost is considered further in the proposed objective function. Furthermore, the proof shows that if the objective function satisfies submodularity, greedy algorithms can achieve $\frac{1}{2}(1-\frac{1}{e})\overline{OPT}$ guarantees.

The proposed approach is illustrated in Fig. 1(a). There are 72 subgoals evenly distributed in the space. A complete graph $\mathcal{G}(V,E)$ is defined, where V represents the subgoal ground set and E represents the edge set. In Fig. 1(b), a minimum spanning tree (MST) of the graph $\mathcal{G}$ can be constructed by Prim's algorithm. The edge cost of the MST is the Euclidean distance between two vertices. Furthermore, a routing cost function is obtained from the compressed sensing technique [8]. In Fig. 1(c), based on the subgoals selected by the CASMO algorithm, the robot searches the space with the maximal coverage under the routing and computational constraints.

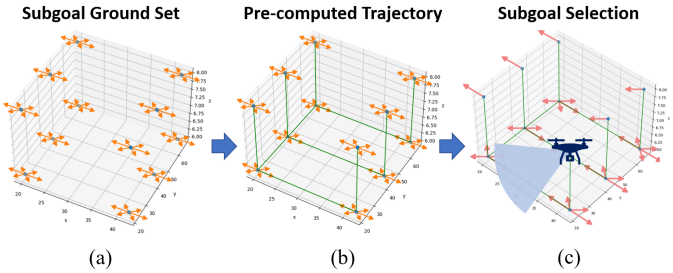


Fig. 1: Illustration of the proposed method. (a) An example of a 3D space with 72 subgoals consisting of subgoal locations (blue dots) and the robot heading (orange arrows). (b) The routing trajectory (green lines) is constructed by MST. (c) The selected subgoals (pink arrows) are obtained by the CASMO algorithm. The robot visits the subgoals and finds the targets in the 3D space.

Some assumptions are made. First, a set of subgoals is given in the beginning. Second, the perception of the robot encompasses inherent uncertainty, while the environment contains occluded targets. Third, the coverage at every subgoal can be pre-computed before the search.

The contributions of this research are as follows: First, the proposed CASMO algorithm incorporates a submodular cost function, consisting of the path cost and the computational cost. Second, since the cost function satisfies submodularity, greedy algorithms achieve $\frac{1}{2}(1-\frac{1}{e})\overline{OPT}$ guarantees. Third, the experiment results show that the proposed method outperforms state-of-the-art approaches in multiple objects search.

This paper is organized as follows. Section II reviews the relevant works on target search methods and submodularity. Section III describes the background knowledge of this research. Section IV introduces the problem formulation. Section V describes the search algorithm. Section VI describes the experiments and analyzes the results. Finally, Section VII draws conclusions and describes future work.

[1]Yan-Shuo Li is a graduate student in the Mathematics Department at National Central University, Taiwan. yanshuolee102@gmail.com
[2]Kuo-Shih Tseng is with the Faculty of the Mathematics Department at National Central University, Taiwan. kuoshih@math.ncu.edu.tw

II. Related work

In this section, the recent work on target search methods and submodularity is reviewed.

A. Target Search

There are a variety of search methods such as the frontier-based search, the probabilistic search, and the coverage approaches.

The frontier-based search method uses the boundary between the explored and unexplored space to explore environments. In [9], the researchers propose a hierarchical architecture for rescue robots. It extends the frontier-based exploration approach in urban search and rescue (USAR) environments to cluttered rubble-filled environments by incorporating terrain information. However, this system is semi-autonomous and requires human assistance.

Due to sensor uncertainty, the probabilistic search methods are applied. Mohamed et al. [10] present the person search-orienting problem (PSOP). The user activity probability density function is used to generate a search plan for the maximum expected detection.

Many search methods constrain the search space in 2D. Zheng et al. [7] present a multi-object search method in 3D environments with a frustum-shaped field of view, which can be applied to mobile robots or drones.

B. Submodularity

Set functions with diminishing return property are submodular functions [11]. Various applications including sensor placement, viral marketing, and robotic search problems can be reformulated as submodular maximization problems with different constraints (e.g., cardinality [11], additive budget [12], and routing [6]).

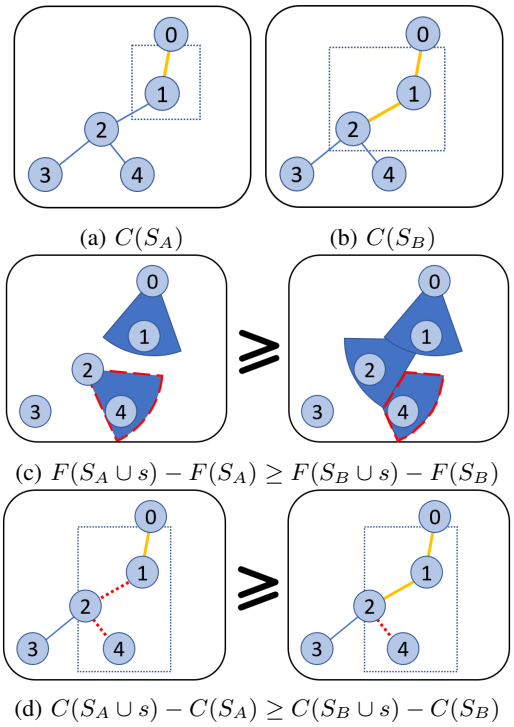
Although these problems are NP-hard, greedy algorithms can find solutions with theoretical guarantees [11]. In sensor placement problems [1] [13], a set of sensors is selected for environmental monitoring to maximize the quantity of information.

Under routing constraints, Zhang et al. [6] propose a generalized cost-benefit (GCB) algorithm to solve the problems. Both maximizing the values of submodular functions and minimizing the path cost of routing functions are NP-hard. Hence, by utilizing TSP-approximation algorithms such as nearest neighbor (NN), GCB finds solutions over the $\frac{1}{2}(1-\frac{1}{e})\widetilde{OPT}$ optimum, where $\widetilde{OPT}$ is approximately optimal. Thus, there is a gap between $\widetilde{OPT}$ and the optimal solution (OPT). To improve the theoretical guarantee of GCB, Lin et al. [5] propose Tree-Structured Fourier Supports Set (TS-FSS) algorithms to boost the theoretical guarantees to the $\frac{1}{2}(1-\frac{1}{e})\overline{OPT}$ optimum, where $\widetilde{OPT} \leq \overline{OPT}$.

III. Background

The background of submodularity and its application in the Fourier domain is introduced as follows.

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



(a) $C(S_A)$ (b) $C(S_B)$

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

(d) $C(S_A \cup s) - C(S_A) \geq C(S_B \cup s) - C(S_B)$

Fig. 2: Illustration of submodularity under the coverage and tree-structured graph. The nodes with decimal numbers represent the geographical locations of the sensors. Solid yellow and blue lines represent the selected and unselected routing paths, respectively. (a) $C(S_A)$ is the routing length of selected nodes S_A, where $S_A = \{0, 1\}$. (b) $C(S_B)$ is the routing length of selected nodes S_B, where $S_B = \{0, 1, 2\}$. (c) In coverage cases, the marginal gains of the coverage function F are marked as the red dashed lines when $s = \{2\}$ is added to the current set $S_A = \{0\}$ and $S_B = \{0, 1\}$. (d) In routing cases, the marginal gains of the routing function C are marked as the red dashed lines when $s = \{4\}$ is added to the current set.

Fig. 2 illustrates the submodularity of the coverage and routing cost functions. There are three sets: the ground set $S = \{0, 1, 2, 3, 4\}$ and two selected sets $S_A = \{0, 1\}$, $S_B = \{0, 1, 2\}$, where $S_A \subseteq S_B \subseteq S$. If the submodular function f is defined as the coverage function F, $F(S_A)$ represents the covered area of the field of view (FOV) when selecting the set S_A (see Fig. 2 (c)). When f is defined as the routing cost function C, $C(S_A)$ is the path length of traversing all elements in S_A (see Fig. 2 (d)). As shown in Fig. 2(b), starting from the root, node 0, the trajectory of $S_B = \{0, 1, 2\}$ is $0 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 0$. That is, $C(S_B) = 2(D(0, 1) + D(1, 2))$, where $D : \mathbb{R}^2 \rightarrow \mathbb{R}$ is a distance measurement such as the Euclidean distance.

In coverage cases, two original sets are defined as $S_A = \{0\}$ and $S_B = \{0, 1\}$. Fig. 2(c) shows that the marginal gain of the coverage function F is the additional covered area (red dashed line) when a new set $s = \{2\}$ is added to the original set. In routing cases, Fig. 2(d) shows that the marginal gain is the extra path length (red dashed line) when a new set s

is added to the original set. Since $S_A \subseteq S_B \subseteq S$, by Def. 1, Fig. 2(c) and (d) show that the marginal gain of S_A will be greater than S_B in the coverage and routing cases. Hence, these two functions are submodular.

Some properties of submodular functions are introduced as follows:

Lemma 1: (Linear combination of submodular functions) [11] If $g(S) = \sum_{i=1}^{n} \alpha_i f_i(S)$, where $f_i : 2^S \to \mathbb{R}^+$ is a submodular function and α_i is a non-negative coefficient, then $g : 2^S \to \mathbb{R}^+$ is also a submodular function.

Corollary 1: (Submodularity of minimum spanning trees) [14] Routing cost $C_r(S) : S \to \mathbb{R}^+$ for the minimum spanning tree (MST) is submodular.

Definition 2: (Fourier transform of submodular functions) [4] Given a submodular function F, its Fourier transform coefficients f and a Hadamard transform matrix H, the inverse Fourier transform equation is $F = H \cdot f$.

In the learning phase, given sampling submodular function values F_M and a reconstruction matrix Ψ, the Fourier coefficients $\hat{f_B}$ can be computed as follows

$$\hat{f_B} = \arg\min_{f_B} \frac{1}{2} \left\| F_M - \Psi f_B \right\|^2 + \lambda \left\| f_B \right\|_1,$$

where λ is the sparsity parameter.

In the reconstruction phase, given the estimated Fourier coefficient $\hat{f_B}$ and the sampling Hadamard matrix ψ_B^f, the reconstruction function is

$$F(S) = \sum_{B^f \in 2^n} \hat{f_B} \psi_B^f(S),$$

where $\Psi\left[i,j\right]_B = \psi_{B_j^f}(A_i) = (-1)^{|A_i \cap B_j^f|}$, A is the input set and B^f is the basis set.

Theorem 1: (Lower bound of GCB under a tree-structured graph) [5] Given a submodular monotone set function F and a cost function c of a tree-structured graph, the GCB approach is to maximize F subject to the cost constraint B. The performance of the set X obtained by GCB is

$$F(X) \geq \frac{1}{2}\left(1 - \frac{1}{e}\right) F(\bar{X}),$$

where

$$\bar{X} = \arg\max_X \{F(X) | c(X) \leq B(1 - \kappa_c + \frac{\kappa_c}{K_c})\},$$

$\kappa_c \in [0,1]$ is the total curvature of the cost function c and K_c describes the solution with the largest set size that meets the constraint.

IV. Problem Formulation

To formulate computation-aware multi-object search problems, the submodularity is used in this research. The objective function is to find a set of subgoals that maximize environmental coverage subject to a cost function. Mathematically,

$$\begin{aligned} \max_{S} \quad & F(S) \\ \text{s.t.} \quad & C(S) \leq B, \end{aligned} \tag{1}$$

where $X = \{1, 2, ..., N\}$ is the ground set containing N subgoals, $F : 2^N \to \mathbb{R}^+$ is the coverage function, $C : 2^N \to \mathbb{R}^+$ is the computation-aware cost function, and B is the cost budget. In 3D environments, the coverage function F is calculated as the ratio of the number of covered voxels to the total number of environmental voxels. C contains the routing cost C_r and the computational cost C_c. The routing cost C_r can be obtained from the path cost reconstruction algorithm [4], and the computational cost C_c is constant.

Corollary 2: (Linear computation of routing cost) [4] The computational cost C_c of the routing cost function $C_r(S) = \Sigma_{B \in 2^n} \hat{f_B} \psi_B(S)$ is submodular.

Proof: Let $C_r(S) = \sum_{e \in S} w(e)$ be a linear (modular) function. Consider two sets T and N, where $T \subseteq N$. It is obvious that $C_r(T \cup e) - C_r(T) = w(e)$ and $C_r(N \cup e) - C_r(N) = w(e)$.

Hence, a linear (modular) function is submodular. Since our computational time is $\mathcal{O}(b)$ [4], where b is the number of bases. A constant function is a modular function. Thus, $\mathcal{O}(b)$ is submodular. ■

Theorem 2: The computation-aware cost function $C(S) = C_r(S) + C_c(S)$ is submodular.

Proof: To prove that $C(S)$ is submodular, it is shown that the cost of the routing path and computational time are submodular in Cor. 1 and 2, respectively. By Lem. 1, the linear combination of submodular functions is also submodular. Therefore, the cost function C is a submodular function. ■

Since the cost function C satisfies the submodularity, by Thm. 1, the proposed method finds solutions achieving $\frac{1}{2}(1 - \frac{1}{e})\overline{OPT}$.

V. Algorithms

The proposed CASMO algorithm includes three algorithms: TS-FSS [5], GCBPlanner [6], and Path Cost Reconstruction [5]. CASMO constructs an MST from the ground set, finds the Fourier support of the tree, and generates a set of subgoals for the tree traversal.

Alg. 1 describes the CASMO algorithm for multi-object search. In line 1, given a map of the search space, TS-FSS [5] constructs an MST and generates the routing path cost of the MST with the compressed sensing technique [8]. The Fourier basis B^f and the Fourier coefficient c_f are returned by the TS-FSS algorithm [5]. The selected subgoal set G is initialized in line 2. Lines 3-15 are the process of iterative subgoal selection by maximizing the objective function $\frac{f(G \cup X) - f(G)}{c(G \cup X) - c(G)}$, where $c(\cdot) = c_r(\cdot) + c_c(\cdot)$ is the computation-aware cost function composed of the path cost reconstruction algorithm c_r [5] and the computational cost function c_c. The subgoal X^* is chosen if $c(G \cup X^*)$ does not exceed the routing cost budget B and X^* is considered the best subgoal in the current step. The subgoal set V rules out the selected best subgoal. This process continues until there is no subgoal in V. The computational complexity for CASMO is $\mathcal{O}(|V|^2)$.

The robot traverses the selected subgoals in a tree-structured way. For instance, in Fig. 2(a), consider a scenario

where the robot is at node 1 and the next subgoal is at node 4. The robot arrives at node 4 before passing through node 2. After arriving at node 4, the robot detects targets in the environment via the camera and checks if all targets have been found. The process terminates when one of the following criteria occurs: (1) all selected subgoals G are visited, (2) the entire process time exceeds the time constraint, or (3) all targets have been found.

Algorithm 1 Computation-Aware Search for Multiple Objects (CASMO)

Input: V (subgoal ground set), B (routing cost budget), f (coverage function), c_r (path cost reconstruction algorithm), c_c (computational time function), Map (map of an environment), B^f (Fourier support), c_f (Fourier coefficient of the submodular tree)

Output: $G \subseteq V$ (selected subgoal set)

$B^f, c_f \leftarrow$ TS-FSS(Map, V)
$G \leftarrow \emptyset$
while $V \neq \emptyset$ **do**
  for $X \in V$ **do**
    $F \leftarrow f(G \cup X) - f(G)$
    $C_r \leftarrow c_r(B^f, c_f, G \cup X) - c_r(B^f, c_f, G)$
    $C_C \leftarrow c_c(G \cup X) - c_c(G)$
    $M \leftarrow F/(C_r + C_C)$
  end for
  $X^* \in \arg\max_X M$
  if $c(G \cup X^*) \leq B$ **then**
    $G \leftarrow G \cup X^*$
  end if
  $V \leftarrow V \setminus X^*$
end while

VI. Experiments

The proposed algorithms (CASMO) and the benchmark algorithms are evaluated according to the average number of detected objects and the expected time to detection (ETTD). Benchmark approaches are state-of-the-art approaches (e.g. GCB-TSP, MR-POUCT, and the other six approaches) [6] [7]. 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 $j \in N$. If the search time exceeds the time constraint, the detection time of the remaining non-detected objects will be set to the maximal time constraint.

The search process is as follows: CASMO generates K subgoals before the drone takes off. Once the drone reaches the first subgoal, it hovers and detects the target within 3 seconds. The UAV then moves to the next subgoal. The process repeats until all objects in the environment have been found, or it arrives at the K^{th} subgoal. Finally, the drone lands on the ground when the search is terminated.

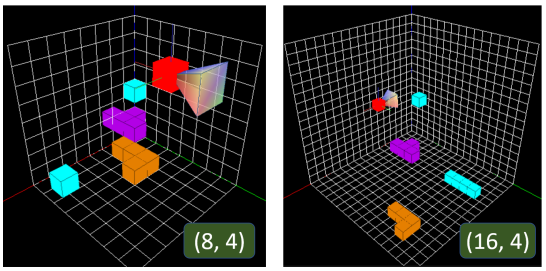


Fig. 3: 3D environments for simulations. The environment sizes of 8 and 16 are evaluated. A tuple (e, t) represents a pair of space size and number of targets. For example, $(8, 4)$ stands for the size of 8^3 with 4 objects within this environment. The red cube represents the robot searching for the target.

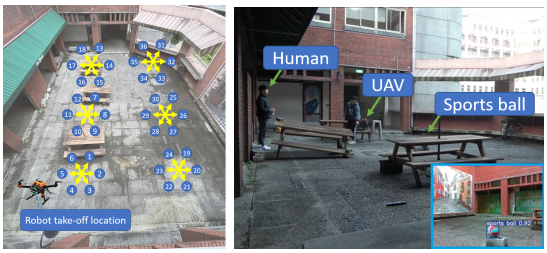
A. Experiment Setup

There are two experiments: simulations and UAV experiments. In the simulations (EX1), the scalability of the proposed approach is tested in various environments. In the UAV experiments (EX2), the performance of the proposed method is examined in a realistic complex environment.

In the EX1, the simulated environment is adopted from [7] (see Fig. 3). There are two different sizes of the environment $E = \{8, 16\}$, and different numbers of targets $T = \{2, 4, 6\}$ in 3D spaces. Each (e, t) pair generates 40 trials with randomly located targets, where $e \in E$ is an $e \times e \times e$ search space, and $t \in T$ is the number of targets that the robot can find in e. The goal of the robot is to find all targets, which contain partial occlusion.

In the search process, the robot can take one of the actions from $A = \{Move(i), Look(j), Detect\}$, where $i \in \{+x, -x, +y, -y, +z, -z\}$ and $j \in \{(\theta_x, \theta_y, \theta_z)\}$. The $Move$ action takes one unit step in the direction of i. The $Look$ action rotates the robot by specifying the angle on the x-y-z axis. The $Detect$ action performs an object detection. The time budget (planning time + moving time) for the whole process is constrained to 20 minutes. Once all the targets have been found or the search time exceeds the constraint, the task is terminated. The experiment parameters are shown in Table I. Notice that the sensing range in this research is smaller than that in [7] since the realistic ratio of the environment size to the camera FOV range is considered. The routing cost (C_r) is calculated by the Manhattan distance between two locations. For example, for any two connected nodes in MST, $v_1 = [x_1, y_1, z_1, \theta_1]$ and $v_2 = [x_2, y_2, z_2, \theta_2]$, the routing cost $C_r(\{v_1, v_2\}) = |x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2| + |\theta_1 - \theta_2|$.

In the EX2, the search environment is a 13×9 m public area on the third floor of the General Education Building at the National Central University. The map and subgoals are shown in Fig. 4(a). The space is divided into voxels whose unit size is $15 \times 15 \times 15$ cm. The goal is to find a human and a sports ball in the environment, shown in Fig. 4(b). Two targets are randomly located and can be partially occluded in the complex environment with obstacles. The searcher is the



(a) (b)

Fig. 4: (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 the UAV searching for two targets (a human and a sports ball).

drone developed by Taiwan Drone 100 shown in Fig. 5. The drone is equipped with NVIDIA Jetson Xavier NX and two Intel RealSense cameras. The Intel RealSense D435i camera is used to explore environments, while the T265 camera is used to localize the drone. Parameters of the D435i camera and the drone are shown in Table I.

TABLE I: Parameters of EX1 and EX2.

Parameters	EX1	EX2
Camera Range	2 grids	$3m$.
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

The YOLOv5 [15] is adopted to detect objects running on NVIDIA Jetson Xavier NX. Moreover, since the targets in the environment may be occluded, the uncertainty of detection is considered. To mark a target as detected, it must surpass a minimum probability threshold of 0.6.

TABLE II: Average number of detected objects in environment sizes $E = \{8, 16\}$. Different numbers of targets $T = \{2, 4, 6\}$ in the 3D space are examined.

Method	8x8x8 Space			16x16x16 Space		
	2	4	6	2	4	6
Exhaustive	0.53	0.95	1.38	0.18	0.25	0.3
MR-POUCT	0.15	0.08	0.3	0.08	0.08	0.18
Options+POUCT	0.2	0.2	0.55	0.05	0.05	0.2
POMCP	0.03	0.05	0.08	0.03	0.03	0.1
Porollout	0	0.03	0.05	0.05	0.03	0.03
POUCT	0.25	0.25	0.3	0.03	0.15	0.22
Random	0.03	0.1	0.3	0.08	0	0.03
GCB-TSP	0.75	1.4	1.65	0.33	0.78	1.15
CASMO	**1.25**	**2.1**	**3.43**	**0.45**	**1**	**1.45**

TABLE III: Expected time to detection (ETTD) in environment sizes $E = \{8, 16\}$. Different numbers of targets $T = \{2, 4, 6\}$ in the 3D space are examined. The unit is minutes.

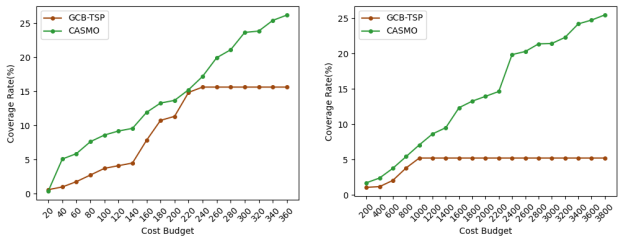
Method	8x8x8 Space			16x16x16 Space		
	2	4	6	2	4	6
Exhaustive	16.15	16.98	17.46	18.83	19.18	19.49
MR-POUCT	18.66	19.68	19.15	19.4	19.65	19.52
Options+POUCT	18.35	19.17	18.49	19.59	19.82	19.43
POMCP	19.82	19.77	19.82	19.77	19.88	19.74
Porollout	20	19.9	19.87	19.6	19.89	19.93
POUCT	17.85	18.92	19.18	19.8	19.4	19.36
Random	19.75	19.59	19.17	19.3	20	19.96
GCB-TSP	13.54	13.96	15.18	**17.56**	**17.31**	**17.4**
CASMO	**11.79**	**13.27**	**12.83**	18.08	17.8	17.73

Fig. 5: A customized UAV developed by Taiwan Drone 100.

B. EX1: Simulation Evaluation

Table II and Table III show the results in simulated environments. CASMO is compared with state-of-the-art baselines (e.g., POMCP, POUCT, Options+POUCT, and porollout) [7].

Table II shows that the drone finds more targets faster in a small environment ($e = 8$) than in a large one ($e = 16$). In addition, as the number of targets decreases in an environment, the difficulty of searching increases. Despite these challenges, CASMO finds the highest number of targets in both environment sizes in comparison with all benchmarks. The advantage of the proposed approach, in contrast with MR-POUCT [7], is that CASMO optimizes the coverage



(a) Coverage rate of $e = 8$ (b) Coverage rate of $e = 16$

Fig. 6: Coverage rate with respect to cost budget constraints in $e = 8$ and $e = 16$.

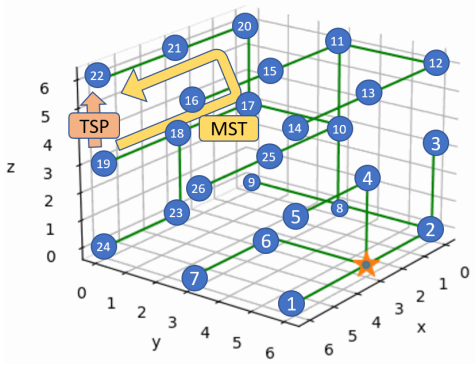


Fig. 7: Minimum spanning tree in a 3D environment of $e = 8$. Each node with a number is a subgoal for the robot to search target. The orange and yellow arrows show the robot trajectory from node 19 to node 22 with TSP and MST, respectively.

function and does not need to replan the search strategy while the environment remains the same. On the other hand, MR-POUCT [7] chooses the maximal value from the search tree and plans an action through observation at each time step.

Table III demonstrates the ETTD of all approaches. Note that all methods are constrained to search within 20 minutes. In $e = 8$, CASMO is the most efficient approach to search in the space. However, in $e = 16$, CASMO outperforms state-of-the-art approaches except for the GCB-TSP [6], which adopts the nearest neighbor algorithm to find the solution of TSP.

To investigate the results, the coverage and routing path of CASMO and GCB-TSP [6] are shown in Fig. 6 and Fig. 7, respectively. CASMO is able to search for an area with larger coverage than GCB-TSP [6] under different budgets. In Fig. 6(a), CASMO reaches a coverage rate of 25% in comparison with the original GCB's coverage rate of 15% in $e = 8$. In Fig. 6(b), CASMO reaches a coverage rate of 26% while GCB-TSP [6] only covers 5% of the environment. These coverage results demonstrate that the coverage of CASMO is better than that of GCB-TSP [6] as Thm. 1 claims. Furthermore, the coverage rate of GCB-TSP [6] converges after some budgets. Since solving the TSP takes much time, the robot is unable to visit all selected subgoals within the time constraint (20 minutes).

However, the better coverage results do not represent the search efficiency. The objective function is to maximize the coverage subject to a budget constraint. The search behavior is to explore the environment within the budget time instead of reducing the robot search time. Furthermore, CASMO traversal is in a tree structure, which confines the routes. For instance, in Fig. 7, if the robot is at node 19 and the next subgoal is at node 22, GCB-TSP [6] will go directly to the destination (the orange arrow) while CASMO will traverse the subtree to node 22 (the yellow arrow).

These results demonstrate that CASMO is able to find the most objects among state-of-the-art approaches since the coverage of CASMO is greater than other approaches. CASMO can outperform other state-of-the-art approaches except GCB-TSP [6] in ETTD due to restrictions on tree traversal.

C. EX2: UAV Experiments

The CASMO, GCB-TSP [6], and MR-POUCT [7] approaches are evaluated in the real environment. Two objects are randomly placed in 5 locations. Experiments are conducted 50 times (5 locations $\times$ 10 trials) for each approach.

Table IV shows the ETTD results for CASMO, GCB-TSP [6], and MR-POUCT [7]. CASMO is the fastest approach among other approaches. Furthermore, the coverage of CASMO is 66.5%, which is the largest among these approaches. These experiments demonstrate that CASMO outperforms the benchmark algorithms.

The summaries of these experiments are as follows. Due to theoretical guarantees of submodularity, submodular approaches (e.g., CASMO, GCB-TSP [6]) outperform state-of-the-art approaches (e.g., MR-POUCT [7]). Due to the boost guarantees of CASMO, CASMO outperforms GCB-TSP [6] in coverage and the average number of detected objects. Due to the traversal of the subtrees, CASMO could not always outperform GCB-TSP [6] in the ETTD since the objective function (Eq. (1)) is to maximize the coverage.

TABLE IV: The expected time to detection (ETTD) of three different approaches.

	CASMO	GCB-TSP	MR-POUCT
Mean (sec.)	**332**	349	594
Std.	28.6	26.2	8.5
Coverage rate	**66.5%**	62.8%	27.3%

VII. Conclusions

This research proposes a CASMO algorithm, which is to maximize the coverage function with the computation-aware cost constraint. Since the objective function and constraint are submodular, the CASMO achieves a $\frac{1}{2}(1 - \frac{1}{e})\overline{OPT}$ guarantee. The experiment results show that CASMO outperforms state-of-the-art approaches and improves computational efficiency.

The future work of this research is as follows. First, the current approach is in known environments. The search for targets in unknown environments will be a potential direction. Second, this research only considers one robot for search. Extending the proposed approach to multi-robot search is another direction. Third, according to the experiment results, traversing a tree structure is inefficient. Generating different MST structures is a way to further improve search efficiency.

Acknowledgment

This research was completed thanks to the financial support from MOST Grant 111-2221-E-008-097, Taiwan.

REFERENCES

[1] A. Krause and C. Guestrin, "Near-optimal observation selection using submodular functions," *Association for the Advancement of Artificial Intelligence (AAAI)*, vol. 7, pp. 1650–1654, 2007.

[2] F. Chen, Y. Lu, Y. Li, and X. Xie, "Real-time active detection of targets and path planning using uavs," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 391–397, 2021.

[3] X. Chen and S. Lee, "Visual search of an object in cluttered environments for robotic errand service," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4060–4065, 2013.

[4] K.-S. Tseng and B. Mettler, "Near-optimal probabilistic search using spatial fourier sparse set," *Autonomous Robots*, vol. 42, no. 2, pp. 329–351, 2018.

[5] P.-T. Lin and K.-S. Tseng, "Improvement of submodular maximization problems with routing constraints via submodularity and fourier sparsity," *IEEE Robotics and Automation Letters*, pp. 1–8, 2023.

[6] H. Zhang and Y. Vorobeychik, "Submodular optimization with routing constraints," *Association for the Advancement of Artificial Intelligence (AAAI)*, vol. 30, no. 1, 2016.

[7] K. Zheng, Y. Sung, G. Konidaris, and S. Tellex, "Multi-resolution pomdp planning for multi-object search in 3d," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2022–2029, 2021.

[8] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[9] B. Doroodgar, Y. Liu, and G. Nejat, "A learning-based semi-autonomous controller for robotic exploration of unknown disaster scenes while searching for victims," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2719–2732, 2014.

[10] S. C. Mohamed, S. Rajaratnam, S. T. Hong, and G. Nejat, "Person finding: An autonomous robot search method for finding multiple dynamic users in human-centered environments," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 1, pp. 433–449, 2020.

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

[12] S. Khuller, A. Moss, and J. S. Naor, "The budgeted maximum coverage problem," *Information processing letters*, vol. 70, no. 1, pp. 39–45, 1999.

[13] A. Dhariwal, B. Zhang, B. Stauffer, C. Oberg, G. S. Sukhatme, D. A. Caron, and A. A. Requicha, "Networked aquatic microbial observing system," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4285–4287, 2006.

[14] Y. T. Herer, "Submodularity and the traveling salesman problem," *European journal of operational research*, vol. 114, no. 3, pp. 489–508, 1999.

[15] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation," Nov. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7347926