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

國立中央大學圖書館學位論文授權書

填單日期：2023 / 09 / 25　　2019.9 版

授權人姓名	邱彥翔	學 號	109221013
系所名稱	數學	學位類別	☑碩士 □博士
論文名稱	Informative path planning via cost-bounded spanning tree	指導教授	曾國師

學位論文網路公開授權

授權本人撰寫之學位論文全文電子檔：

・在「**國立中央大學圖書館博碩士論文系統**」.

(　)同意立即網路公開

(✓)同意 於西元2025年09月25日網路公開

(　)不同意網路公開，原因是：＿＿＿＿＿＿＿＿

・在**國家圖書館「臺灣博碩士論文知識加值系統」**

(　)同意立即網路公開

(✓)同意 於西元2025年09月25日網路公開

(　)不同意網路公開，原因是：＿＿＿＿＿＿＿＿

依著作權法規定，非專屬、無償授權國立中央大學、台灣聯合大學系統與國家圖書館，不限地域、時間與次數，以文件、錄影帶、錄音帶、光碟、微縮、數位化或其他方式將上列授權標的基於非營利目的進行重製。

學位論文紙本延後公開申請 (紙本學位論文立即公開者此欄免填)

本人撰寫之學位論文紙本因以下原因將延後公開

・**延後原因**

(　)已申請專利並檢附證明，專利申請案號：

(✓)準備以上列論文投稿期刊

(　)涉國家機密

(　)依法不得提供，請說明：＿＿＿＿＿＿

・**公開日期**：西元2025年09月25日

※繳交教務處註冊組之紙本論文(送繳國家圖書館)若不立即公開，請加填「國家圖書館學位論文延後公開申請書」

研究生簽名：邱彥翔　　指導教授簽名：曾國師

*本授權書請完整填寫並親筆簽名後，裝訂於論文**封面之次頁**。

國家圖書館學位論文延後公開申請書

Application for Embargo of Thesis/Dissertation

申請日期：民國 112 年 7 月 25 日

Application Date: 2023 / 07 / 25 (YYYY/MM/DD)

申請人姓名 Applicant Name	邱瑄翔	**學位類別** Graduate Degree	☑碩士 Master ☐博士 Doctor	**畢業年月** Graduation Date (YYYY/MM)	民國 112 年 07 月
學校名稱 University	國立中央大學		**系所名稱** School/Department	數學	
論文名稱 Thesis / Dissertation Title	Informative path planning via cost-benefit spanning tree				
延後公開原因 Reason for embargo	☑涉及機密 論文投稿 Contains information pertaining to the secret. ☐專利事項，申請案號： Filing for patent registration. Registration number: ☐依法不得提供，請說明： Withheld according to the law. Please specify.				
申請項目 Options	☑紙本論文延後公開 Delay public access to the printed copies of my thesis, but leave the online bibliographic record open to the public.			☐書目資料延後公開 Delay public access to online bibliographic record of my thesis.	
公開日期 Delayed Until	民國 114 年 7 月 25 日 2025 / 07 / 25 (YYYY/MM/DD)			☐不公開 Prohibited from public access.	

申請人簽名：Applicant Signature: 邱瑄翔

指導教授簽名：Advisor Signature: 曾國邦

學校認定/審議單位章戳：Seal of the Authorization Institute:

國立中央大學 系學數 印

【說明】

1. 依教育部107年12月5日臺教高(二)字第1070210758號函及109年3月13日臺教高通字第1090027810號函，請據實填寫本申請書並檢附由學校認定或審議單位認定之證明文件，經由學校向本館提出申請，無認定或審議單位章戳者退回學校處理。
2. 論文尚未送交國家圖書館，請於提送論文時，夾附親筆簽名申請書1份。
3. 論文已送達國家圖書館，請將親筆簽名申請書一式2份掛號郵寄10001臺北市中山南路20號國家圖書館館藏發展及書目管理組，並於信封註明「學位論文延後公開申請書」。
4. 本館保存之學位論文依學位授予法應提供公眾於館內閱覽紙本，或透過獨立設備讀取電子資料檔，二者依表單填寫日期公開。

【Notes】

1. Please fill in all blanks and attach the certification documents approved by the university and apply through the university. The application form will not be accepted for processing until all information, signatures, and stamps are included.
2. If the thesis or dissertation is not yet submitted to the NCL, please attach the signed application form to the thesis or dissertation.
3. If the thesis or dissertation has been submitted to the NCL, please send a registered letter with 2 copies of the signed application form attached. The letter should be addressed to "Collection Development Division", National Central Library with a note in the envelope indicating "Application for delay of public release" to the following address. No.20, Zhongshan S. Rd., Zhongzheng District, Taipei City 10001, Taiwan (R.O.C.)
4. The delayed date of printed copies and the independent viewing equipment will synchronize.

(**申請者免填**，以下由國家圖書館填寫 For Internal Use)

承辦單位_館藏組：_____ 日期/處理狀況：

典藏地：_____ 登錄號：_____ 索書號：

會辦單位_知服組：_____ 日期：_____ ☐移送並註記，原上架日期：

論文系統：_____ 日期：

國立中央大學碩士班研究生
論文指導教授推薦書

數學系碩士班 學系/研究所 邱韋翔 研究生

所提之論文 Informative path planning using cost-benefit spanning tree

係由本人指導撰述，同意提付審查。

指導教授 曾國邦 (簽章)

112 年 6 月 27 日

1120627

國立中央大學碩士班研究生
論文口試委員審定書

數學系碩士班 學系/研究所 邱韋翔 研究生

所提之論文 Informative path planning using cost-benefit spanning tree

經由委員會審議，認定符合碩士資格標準。

學位考試委員會召集人 方煒

委員 許正雄

鄭紹濠

曾國師

中華民國 112 年 7 月 12 日

1120630

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

The challenge of IPP is to find the optimal path that maximizes the information subject to budget constraints. However, finding the optimal path is a NP-hard problem. To solve these problems, some methods are proposed. In [1] [2], the authors proposed an informative path planning framework in online settings with adaptivity requirements. This approach enables agents to find a target based on the given constraints. However, these approaches cannot achieve the theoretical guarantees.

Reformulating IPP problems as submodular maximization problems is a promising approach with theoretical guarantees [3]. If the IPP problems can be reformulated as a maximizing submodular function subject to some constraints (e.g. cardinality [3], additive budget [4], and routing [5]), the variant greedy algorithms can give theoretical guarantees [3] [6].

The generalized cost-benefit (GCB) is proposed and proves the theoretical guarantees in the routing constraints [5], which is a traveling salesman problem (TSP) [7]. Hence, this problem includes two NP-hard problems. First, that the agent finds the maximal information sets K from S sets [3] is a set-covering problem [8]. Second, that the agent finds the least route from K subgoals is a TSP [9]. The GCB algorithm achieves $\frac{1}{2}(1-\frac{1}{e})\widetilde{OPT}$, where $\widetilde{OPT}$ is the approximation of optimum from the overestimated routing cost.

Since it is infeasible to find the least route from K subgoals, the approximated algorithms are adopted for GCB. However, the approximated algorithms are overestimated. It causes the GCB algorithm to terminate before utilizing all budgets. The GCB-MST utilizes the submodularity of the spanning trees to boost the theoretical guarantee [10]. The GCB and the GCB-MST achieve $\frac{1}{2}(1-\frac{1}{e})\widetilde{OPT}$ and $\frac{1}{2}(1-\frac{1}{e})\overline{OPT}$, respectively, where $\widetilde{OPT} \leq \overline{OPT} \leq OPT$, $\overline{OPT}$ is the approximation

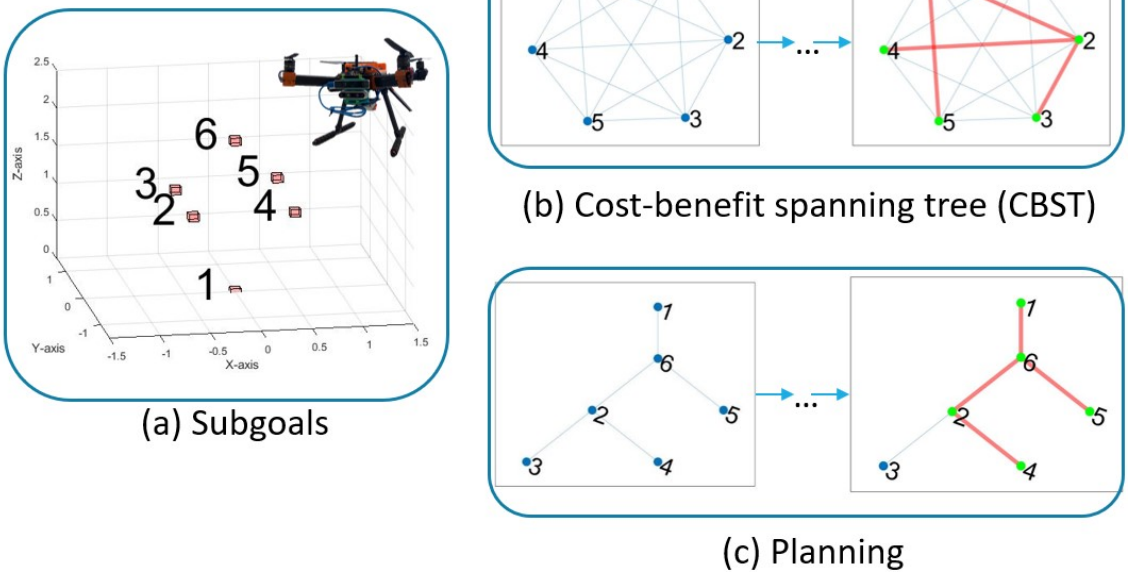


Figure 1.1: Illustration of the proposed method. (a) Subgoals. The blue points and decimal numbers represent subgoals and the index of subgoals, respectively. (b) The cost-benefit spanning tree. The green points, red lines and decimal numbers represent nodes in spanning tree, edges in spanning tree and the index of subgoals, respectively. (c) Path. The green points and red lines represent the path nodes and path edges, respectively.

of optimum from submodular tree-structured graph cost.

The GCB-MST adopts the minimum spanning tree (MST) as the tree structure. However, the MST could not be the best spanning tree. To improve the performance, this research proposes cost-benefit spanning tree (CBST) algorithm, which generates subgoals as a cost-benefit objective. As Fig. 1.1 (a) shows, there are 6 nodes including 1 source node (index 1) in the map. As Fig. 1.1 (b) shows, the approach using cost-benefit algorithm to span the tree. As Fig. 1.1 (c) shows, the agent plans the path via greedy approaches.

The contributions of this research are as follows: First, the informative path planning on terrain is reformulated as a submodular maximization problem with routing constraints. The proposed algorithm, CBST, is able to solve this problem with theoretical guarantees. Second, this research analyzes the performances of the different types of spanning trees. Third, the experiments demonstrate that the proposed approaches outperforms the benchmark.

The paper is organized as follows. Section 2 reviews the relevant

work. Section 3 describes the background knowledge of this research. Section 4 introduces the problem formulation. Section 5 describes the proposed algorithms. Section 6 describes the experiments. Finally, Section 7 reports the conclusions and future work.

2 Related work

The prior works of probabilistic search, informative path planning (IPP), submodular maximization problems and the Prim-Dijkstra algorithm are discussed in this section.

2.1 Probabilistic search

Probabilistic search consists of perception and decision-making [11]. Perception is to estimate where the target is. Bayesian filter enables agents to estimate the probability distribution of the targets [12]. Decision-making is to find the optimal path according to perception. However, finding the optimal solution for this problem is NP-hard [13].

There are two steps in the perception of probabilistic search. First, a probabilistic search is to construct a probabilistic map including the initial information. The probabilistic map is composed of cells. Each cell represents whether the target is located or not. Second, the agent runs the Bayesian filter to update the probabilistic cell of the target existing or not [14] [15].

Occupancy grid maps updated by Bayesian filter are the most commonly used for spatial sensing in perception [16]. In [15] [17], the researchers show searching for one target with different parameters using a Bayesian filter. In [2], the researchers show the an UAV is able to execute terrain monitoring in discrete environments using Bayesian search.

2.2 Informative path planning (IPP)

IPP is to find the optimal path for an agent to maximize the predefined information subject to budget constraints [18]. The IPP problems can be classified by (i) non-adaptive and (ii) adaptive planning strategies. If the agent has information about the environment in advance, and plans the path before taking off, it is called non-adaptive

methods [19], e.g., coverage methods [20] [21], pareto optimization methods [22], evolution algorithm methods [23]. On the other hand, if the agent is allowed to change the path as the information collected during flight, it is called adaptive methods [24], e.g., continuous-space informative path planner (CIPP) method [25], adaptive submodularity with hypothesis pruning methods [26], and non-myopic methods [27].

The IPP problem can be considered as the data gathering mission amounts to one of sequential decision-making under uncertainty, which can be conducted as a Partially Observable Markov Decision Process (POMDP)
[28], which is NP-hard. Although it is NP-hard problems, the greedy approach can obtain near-optimal solutions [3]. In [17], the researchers proposed maximizing the cumulative extended probability of detection. The method can solve the IPP problem with $(1 - \frac{1}{e})$ lower bound guarantee with high probability.

In [29] [2], the authors adopt two-steps approach to adaptive plan strategies. First, the agents find the solutions greedily to maximize the reduction of Shannon's entropy in the map. It is similar to frontier-based approaches for map exploration problems [30]. Second, the agents optimize the subgoals by Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [1] [29]. CMA-ES is an evolutionary approach with generic global optimization [31]. Although the methods in [2] [29] are adaptive, there are no theoretical guarantees.

2.3 Submodular maximization problems

A set function is submodular if it follows the diminishing returns property. If the function is nondecreasing and submodular, greedy policies find the solutions with theoretical guarantees [3]. Various applications include map exploration [32] [33], collecting lake information [27], locating a non-adversarial target [34] [17], and placing sensors for indoor temperature prediction [35]. In [5], the researchers propose generalized cost-benefit (GCB) algorithm for submodular maximization problems with routing constraints. It was proved that the GCB algorithm has $\frac{1}{2}(1 - \frac{1}{e})\widetilde{OPT}$ theoretical guarantees where $\widetilde{OPT} \leq OPT$. In [10], the researchers further improve the guarantees to $\frac{1}{2}(1 - \frac{1}{e})\overline{OPT}$ via the sub-

modularity of routing cost trees [7] and recovering set functions in the Fourier domain [36] [17], where $\widetilde{OPT} \leq \overline{OPT} \leq OPT$.

2.4 Prim-Dijkstra algorithm

The prim algorithm [37] is to solve the Minimum spanning tree (MST) problems while the Dijkstra algorithm [38] is to solve the shortest path tree (SPT) problems. In [39], the researchers combine Prim and Dijkstra (PD) constructions which trade off path length (PL) and total tree weight (TW) to solve routing tree problem, where PL represents the length from the source vertex to the current vertex along the current tree and TW represents the total weight in the tree. In [40], the researchers propose the *PD-II* algorithm via incorporating total detour cost and the amount of suboptimal PL for each node for improving PD algorithm. In [10], the researchers adopt MST as a routing cost tree to improve theoretical guarantees.

3 Background knowledge

The section contains submodularity, the lower bound of GCB under the tree-structured graph, Prim and Dijkstra (PD) algorithm, and extended probability of detection (EPD).

3.1 Submodularity

The definition and illustration of submodularity are as follows:

Definition 1 (Submodularity [3]). *Given a finite set* $S = \{1, 2, ..., N\}$, *a submodular function is a set function* $F : 2^N \to \mathbb{R}$ *that satisfies the diminishing return property. For every* $S_A, S_B \subseteq S$ *with* $S_A \subseteq S_B$ *and every* $s \in S \setminus B$,

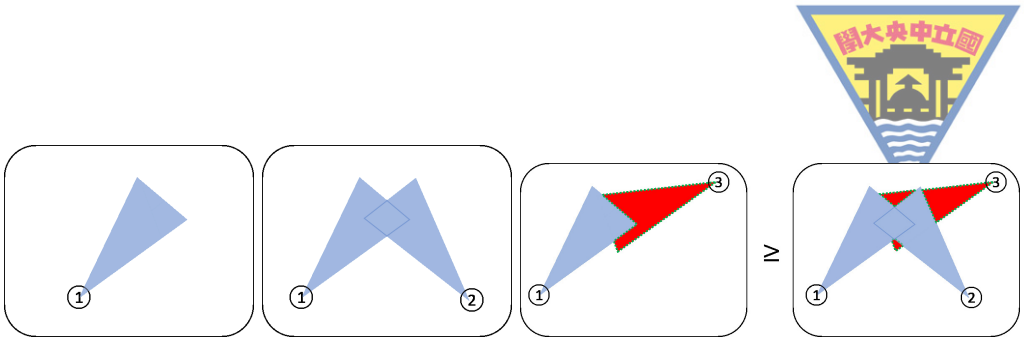
$$F(S_A \cup s) - F(S_A) \geq F(S_B \cup s) - F(S_B) \tag{3.1}$$

holds.

To illustrate the concept of submodularity, an example is shown in Fig. 3.1. There are three ground sets ($S = \{1, 2, 3\}$). $S_A = \{1\}$ and $S_B = \{1, 2\}$ represent the selected two sets, respectively. The set $S_B = \{1, 2\}$ means that the sensors are selected at location ① and ②. $F(S_A)$ and $F(S_B)$ mean the coverage of sensor at location ① and ①② (see Fig. 3.1(a)(b)), respectively. The submodular gain of S_A and S_B after adding a set $s = \{3\}$ is represented by the green dashing lines (see Fig. 3.1(c)). It is clear that the coverage function satisfies the diminishing return property (Eq. 3.1). Alternatively, the objective function of maximizing coverage is submodular. Greedy approaches can generate near-optimal solutions even if maximal coverage is NP-hard problem.

3.2 Lower bound of GCB

To further apply submodularity to routing constraints, some definitions and the lower bound of GCB are introduced as follows:



(a) $F(S_A)$ (b) $F(S_B)$ (c) $F(S_A \cup s) - F(S_A)$ and $F(S_B \cup s) - F(S_B)$

Figure 3.1: Illustration of submodularity. The decimal number represents the selected sensor. The colorful and white areas represent the covered and uncovered areas, respectively. (a) $F(S_A)$ represents the covered area by S_A, where $S_A = \{1\}$. (b) $F(S_B)$ represents the covered area by S_B, where $S_B = \{1, 2\}$. (c) The green dash lines represent the submodular gain after adding s, where $s = \{3\}$. Left figure shows the $F(S_A \cup s) - F(S_A)$ and right figure shows that $F(S_B \cup s) - F(S_B)$.

Definition 2 (Total curvature [41]). *Given a finite set $S = \{1, 2, ..., N\}$, a monotone submodular function f, the total curvature of f is defined as:*

$$\kappa_f = 1 - \min_{s \in S: f(\{s\}) > 0} \frac{f(S) - f(S \setminus \{s\})}{f(\{s\})}. \tag{3.2}$$

If the $\kappa_f = 0$, then f is modular.

Definition 3 (The largest size of feasible solution K_c [5]). *Given a ground set S and cost function c, K_c is defined as:*

$$K_c = \max_{X \subseteq S} \{|X| \ | c(X) \leq B\}, \tag{3.3}$$

where B is budget.

Theorem 3.1 (Lower bound of GCB under the tree-structured graph [10]). *Given a submodular monotone set function f and a tree-structured graph cost function c, the GCB approach is to maximize f subject to the budget B. The performance of a set X achieves*

$$f(X) \geq \frac{1}{2}(1 - \frac{1}{e}) f(\overline{X}), \tag{3.4}$$

where

$$\overline{X} = \arg\max_X \{f(X) | c(X) \leq B(1 - \kappa_c + \frac{\kappa_c}{K_c})\}. \tag{3.5}$$

3.3 Prim and Dijkstra (PD) algorithm

Minimum spanning tree is to minimize the total weight (TW) in a weighted undirected graph. Prim's algorithm is a well-known greedy

algorithm to find a minimum spanning tree for a weighted undirected graph [37].

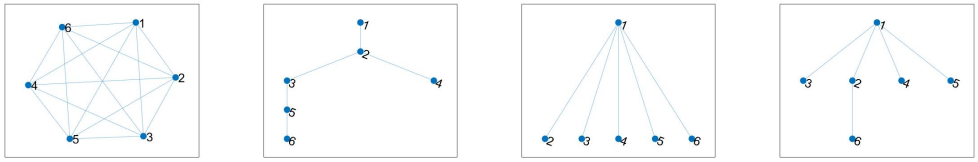
Dijkstra algorithm is a well-known algorithm to find the shortest path between source node and terminal node in an undirected weighted graph. Dijkstra algorithm is adopted for SPT to find the source point between the other nodes in graph. To combine Prim and Dijkstra algorithms, the objective function is proposed as follows:

Definition 4 (The objective function of Prim and Dijkstra algorithm [39]). *The v_i and v_j of Prim and Dijkstra algorithm are chosen to minimize*

$$(\alpha \cdot l_i) + d_{ij} \; s.t. \; v_j \in T, v_i \in V - T, \tag{3.6}$$

where $\alpha \in [0, 1]$ is a tuning parameter, l_i is the length from source (start point) to node v_i, d_{ij} is the distance between v_i and v_j, V is all vertices in graph and T is current growing tree.

Prim-Dijkstra (*PD*) algorithm [39] can generate different types of spanning trees. If $\alpha = 0$, PD algorithm is Prim algorithm; if $\alpha = 1$, PD algorithm is Dijkstra algorithm. As Fig. 3.2(a) shows, there are six subgoals on this map. As Fig. 3.2(b) shows, the Prim algorithm generates the minimum spanning tree (MST). As Fig. 3.2(c) shows, the Dijkstra algorithm on this map generates the shortest path tree. In the *PD* algorithm, different parameters generate different spanning trees.



(a) The complete graph (b) MST ($\alpha = 0$) (c) SPT ($\alpha = 1$) (d) Tree ($\alpha = 0.5$)

Figure 3.2: Illustration of the *PD* algorithm. (a) The complete graph. The blue points, blue edges and decimal numbers represent subgoals, paths and the index of subgoals, respectively. (b) Minimum spanning tree (MST). (c) Shortest path tree (SPT). (d) The tree between MST and SPT. Tuning the α parameter generates different spanning trees.

3.4 Extended probability of detection (EPD)

IPP problems can be formulated as POMDP problems [28], which are NP-hard. Hence, probability of detection model is proposed. However, the assumptions of probability of detection are not realistic, i.e.,

the agent only moves to neighbor cells, and the sensor has no overlapping coveage. The extended probability of detection (EPD) was proposed [42] to apply to real world. The definition of EPD is as follows:

Definition 5 (Extended probability of detection (EPD) [42]). *The agent gets the information z. The assumptions of EPD are as follows:*
(i) There is no target detection ($z = 1$) along the path.
(ii) The sensing overlapping is available.
(iii) The agent could move to any subgoals.
The equation of the cumulative EPD is defined as

$$f_p(S_g) = \sum_{i=1}^{T} P(S_{g,i}) \cdot g \tag{3.7}$$

, where f_p is the cumulative EPD along the path, g is glimpse function and $P(S_{g,i})$ is the probability of covered cells at the i-th subgoal. The value of glimpse function can be calculated from the confusion table.

As Fig. 3.3 shows, the agent's sensing area is not fitting in its local cell. Due to the availability of sensing overlap, the probability of sensing area is recomputed. The assumption is that the agent moves along the path without any detections ($z = 0$). Finally, the agent can navigate to any subgoals instead of neighborhood areas.

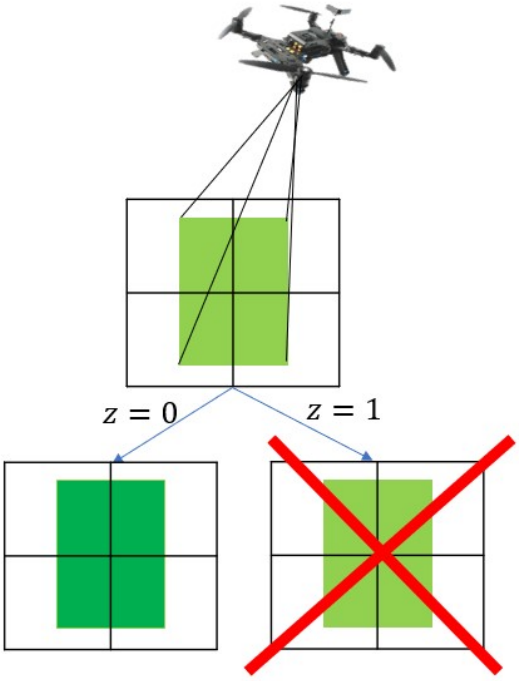


Figure 3.3: Illustration of no detection. In the PD assumption, the agent assumes there is no detection of a target along a path.

4 Problem formulation

The IPP problem is to plan a path where the agent collects information subject to constraints. The major difference between this research and conventional IPP is that the agent collects data above a terrain. The formulation is as follows: Given the ground set S, and $|S| = N$, the goal is to find subgoals that maximize the information subject to the routing constraints.

$$\max f(X) s.t. \quad c(X) \leq B, \tag{4.1}$$

where $X \subseteq S$, f is informative objective function $f : 2^N \to \mathbb{R}^+$, c is the routing cost function $c : 2^N \to \mathbb{R}^+$, and B is the budget. The environment is known, so f and c are oracles that output probability and routing cost values, respectively.

4.1 Cost-benefit spanning tree (CBST)

This problem can be reformulated as a submodular maximization problem. It can achieve theoretical guarantees via GCB approaches. If the routing route is tree structure, the theoretical guarantees will be tighter than GCB [10]. However, how to find an efficient spanning tree structure is an issue.

No matter how the information-rich vertex is far from the source node or not, the agent will go to this point using GCB algorithm by traditional shortest path tree-structured cost function, which is inefficient in search. If the information-rich vertex is far from the source node, the agent may not go through this point using GCB algorithm by MST structured cost function. To solve this issue, the CBST further considers the benefit of f.

The cost-benefit spanning tree is similar to Prim-Dijkstra algorithm. The major difference is that the objective function of CBST

is cost-benefit. The objective function of CBST is

$$\max(\frac{f(v_j|T)}{\alpha \cdot l_i + d_{ij}}), \quad s.t. \quad v_i \in T, v_j \in V - T, \tag{4.2}$$

where f is accumulated extended probability of detection function, and the other notations are the same with Eq. 3.6 [39].

To illustrate the MST and CBST, an example is shown in Fig. 4.1 and 4.2. There are 6 nodes including a source node (index 1) in the map. As Fig. 4.1 shows, the black area has less probability than green area after agent went to v_2 and scanned the area. The scanning areas of v_6 overlaps that of v_2, v_3, v_4, and v_5 since the altitude of v_6 is higher than that of v_2, v_3, v_4, and v_5. The probability of each vertex is $p(v_2) = 0.25, p(v_3) = 0.25, p(v_4) = 0.25, p(v_5) = 0.25$ and $p(v_6) = 1$. Notice that the total probability is over 1 because there is overlapping.

The tree structure of Fig. 4.1 is shown in Fig. 4.2. v_6 is far from v_1 in MST structure (see Fig. 4.2(b)). On the contrary, v_6 is near by v_1 in Fig. 4.2(c) in CBST structure. Under such situations, GCB-CBST performs better than GCB-MST.

On the other hand, the UAV height affects the detection rate. The glimpse function (g) is to describe the relationship between the height and detection rate (see Fig. 4.2(d)). The f in Eq. 4.2 is calculated by $p(v_i)$, path and glimpse function (g), where $i \in \{1, ..., N\}$.

4.2 Theoretical bound of CBST

This section shows the theorems relevant to different kinds of spanning trees. To introduce the theoretical bound of GCB under tree-structured graphs, the relationship between the total curvature κ_c (see Def. 2), the height of tree H, and the number of subgoals are as follows:

According to Thm. 3.1, the tighter theoretical bound is relevant to (κ_c). The κ_c of SPT and MST under constraints are proved as follows:

Theorem 4.1 (κ_c in tree structure [43]). *Given the tree structure of the cost function c, then κ_c either equal to 0 or 1.*

Since the κ_c in tree structure is equal to 1 or 0, the subsequent theorems focus on on exploring the relationship between κ_c and the height of the tree (H).

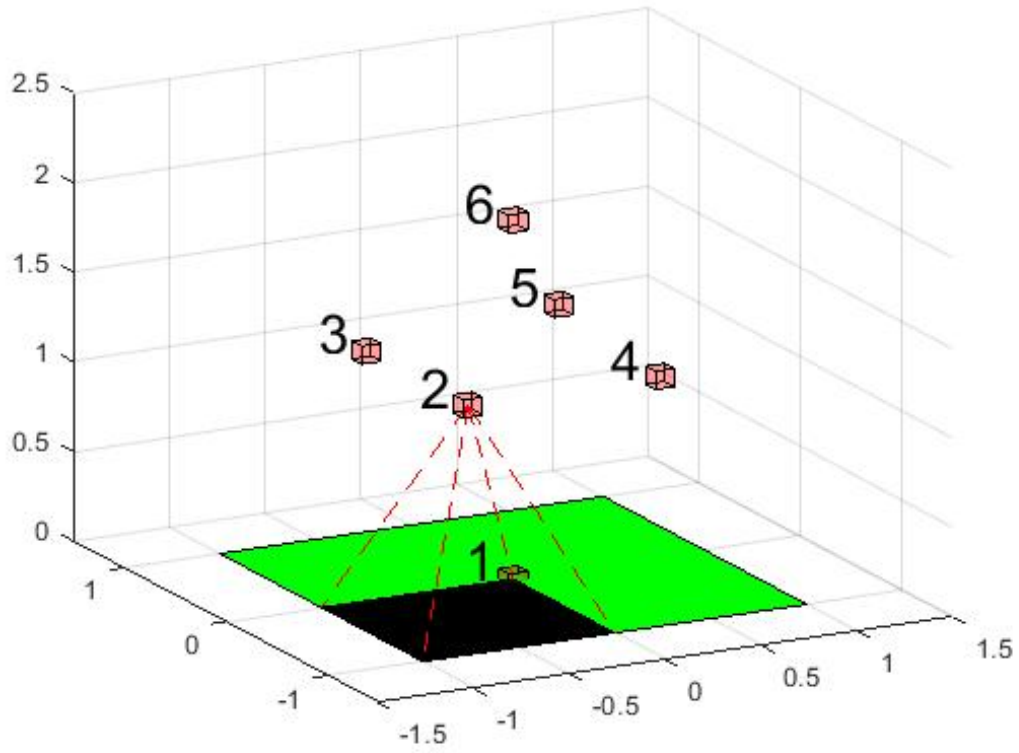
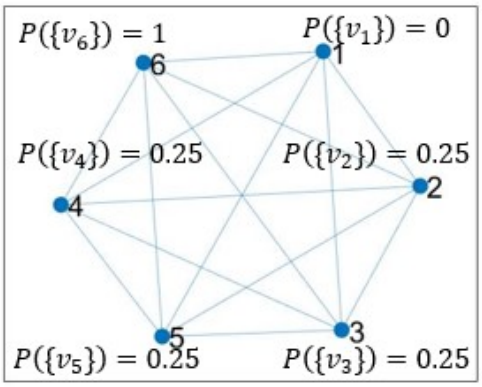
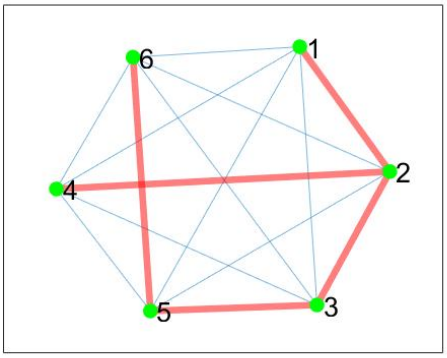


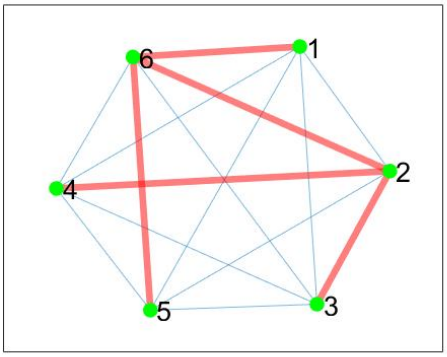
Figure 4.1: Illustration of terrain monitoring. The red cubes and decimal numbers represent subgoals and indexes, respectively. The green and black regions represent lower probability and higher probability, respectively.



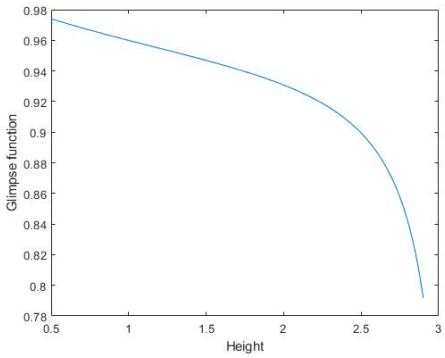
(a) Map and subgoals



(b) Tree structure via MST



(c) Tree structure via CBST



(d) Glimpse function

Figure 4.2: Example of MST and CBST. The vertex 1 is the source. The map is as same as Fig. 3.2 (a). (a) Map and subgoals. The blue points, blue edges, function and decimal numbers represent subgoals, paths, probability of each nodes and the index of subgoals, respectively. (b) Tree structure via MST. (c) Tree structure via CBST. (d) Glimpse function (g).

Theorem 4.2 (The relationship between κ_c and the height of trees)**.** *Given the tree structure of the cost function and the height of the tree (H),*
if $H = 1$, $\kappa_c = 0$*;*
if $H > 1$, $\kappa_c = 1$.

Proof. W.L.O.G., we assume v_1 is the source node.

Case 1: $(H = 1)$
$\because H = 1$.
$\therefore \forall v_i$ are only connecting v_1, where $i \in \{2, ..., n\}$, n is the amount of nodes.
$\Rightarrow c(v_i) = ||v_i - v_1||, i \in \{2, ..., n\}$
$\Rightarrow \kappa_c = 0$ (Eq. 3.2)

Case 2: $(H > 1)$
$\therefore \exists$ the height is equal to 2 or greater than 2.
Since the case of $H > 2$ includes the case of $H = 2$, only the case that the height is 2 will be proved.
W.L.O.G., assume v_2 is the node of depth $= 1$, v_3 is the node of depth $= 2$, and the node between v_1 and v_3 is v_2.
Let $S = \{v_1, v_2, v_3\}$, and $s = \{v_2\}$
$\because c(S) = 2 * (||v_1 - v_2|| + ||v_3 - v_2||)$ and
$c(S \setminus \{s\}) = 2 * ||v_1 - v_3||$
$= 2 * (||v_1 - v_2|| + ||v_3 - v_2||)$
The first equation is by definition, and the second one is by the tree structure.
There is only one path between v_1 and v_3.
$\therefore c(S \setminus \{s\}) = c(S)$
By definition of κ_c (Eq. 3.2),
$\min_{s \in S: f(\{s\}) > 0} \frac{f(S) - f(S \setminus \{s\})}{f(\{s\})} = 0$
$\therefore \kappa_c = 1$

□

Theorem 4.3 (κ_c in SPT)**.** *Given an empty map and the shortest path*

tree structure of the cost function c, then $\kappa_c = 0$.

Proof. Given a complete graph $G = (V, E), V = \{v_1, ..., v_n\}$
$\because \forall i \neq j \neq k \in \{1, ..., n\}, ||v_k - v_i|| + ||v_i - v_j|| \geq ||v_k - v_j||$
the equality only holds in the v_i, v_j, and v_k in a straight line.
$\therefore$The height of shortest spanning tree is 1, i.e. the number of maximal edge between the source node and the others is 1.
$\because$ the height of shortest spanning tree is 1.
$\therefore$ by Thm. 4.2, $\kappa_c = 0$

□

As Fig. 4.3 shows, the vertex 1 is the source node, and the others will be in level 2 due to triangle inequality.

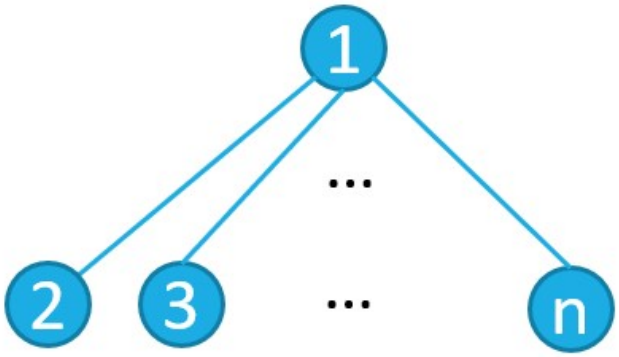
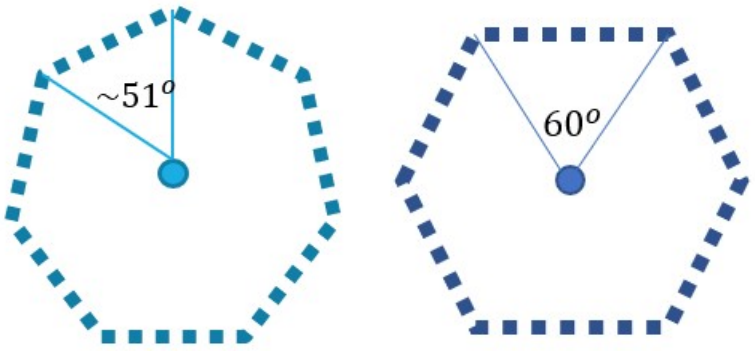


Figure 4.3: Illustration of tree height of Thm. 4.3



(a) Regular heptagon (b) Hexagon

Figure 4.4: Illustration of minimum numbers of Thm. 4.4

To investigate the relationship between κ_c and the number of subgoals, the kissing number problem is introduced for the proof. The kiss-

ing number problem investigates the maximum number of non-overlapping spheres that can touch a central sphere in a given space. It aims to determine the highest possible number of points on the surface of a sphere that can be equidistant from a fixed central point.

Theorem 4.4 (κ_c in MST). *Given an empty map and the minimum spanning tree structure (MST) of the cost function c, the* $\kappa_c = 1$ *when*

$$(i)|S| > 7 \text{ in } 2D \text{ environments}; \tag{4.3}$$

$$(ii)|S| > 10 \text{ in } 3D \text{ environments}. \tag{4.4}$$

Proof. Given complete graph $G = (V, E), V = \{v_1, ..., v_n\}$
W.L.O.G., we assume v_1 is the source node.

Case (i): 2D environments
Proof by contradiction is as follows.
$|S| \leq 7$ in 2D environment.
The subgoals are arranged in the pattern of vertices of a hexagon, and the source node is in the center of a hexagon.

$$\therefore \forall j, k \in \{1, ..., n\}, ||v_1 - v_j|| \leq ||v_j - v_k||. \tag{4.5}$$

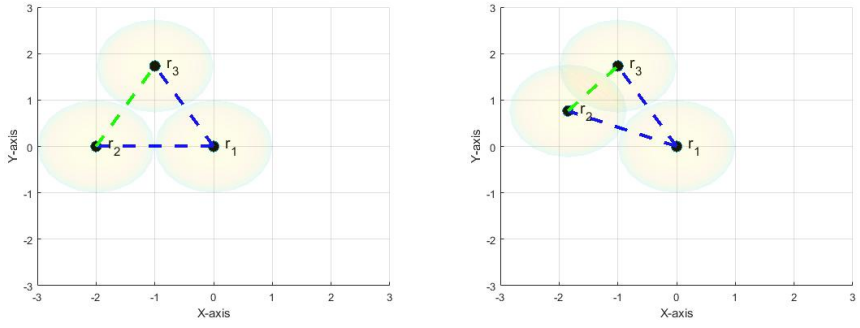
By Eq. 4.5 and the property of MST, the height of MST is equal to 1.
$\Rightarrow \kappa_c = 0$. (by Thm. 4.2)
Hence, $|S| > 7$.
Case (ii): 3D environments
This problem can be seen as
$\max(N)$ s.t. $||v_1 - v_j|| \geq ||v_j - v_k||, j, k \in \{1, ..., N\}$. When $n > N$, it represents $H > 1 \Rightarrow \kappa_c = 1$

It is similar to the kissing number problem. $k(d)$ denotes the highest number of non-overlapping spheres in $\mathbb{R}^d$ that can touch another sphere of the same size. $k(3) = 12$ has been proved [44]. The relationship between the kissing balls and the tree is as follows: As shown in Fig. 4.5 and 4.6, the central sphere (r_1) represents the root of the tree. The highest possible number of points on the surface of a sphere $(r_1, ..., r_N)$

corresponds to the number of points (N) (see Fig. 4.5(a)). The overlapping spheres correspond to the distances between these points (green lines) are smaller than the distances between these points and root (blue lines) (see Fig. 4.5(b)).

The source node is placed on the surface ($x = y = z = 0$), and the other nodes are placed above the surface ($x \in \mathbb{R}, y \in \mathbb{R}$ and $z \geq 0$). First, there are at most six points in xy plane (see Fig. 4.6(a)). Second, since $k(3) = 12$, the kissing number in the upper xy plane and in lower xy plane are 3 and 3, respectively (see Fig. 4.6(b)). It represents $v_2, ..., v_7$ are points in xy plane, and v_8, v_9, v_{10} are upper xy plane. $\because$ There are no overlapping spheres.
$\Rightarrow ||v_1 - v_j|| \leq ||v_j - v_k||, \forall j \neq k \in \{2, ..., 10\}$
Finally, there are six balls in xy plane and three balls above xy plane.
$\Rightarrow$ there are 9 balls satisfying conditions.
$\Rightarrow$ the height of MST is greater than 1 if the number of subgoals greater than 10 (including root) in a three-dimension environment.
$\Rightarrow \kappa_c = 1$.

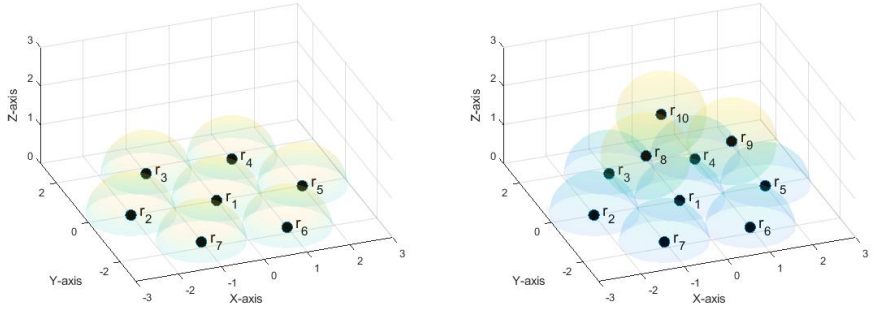
□



(a) Non-overlapping (b) Overlapping

Figure 4.5: Illustration of overlapping balls in Thm. 4.4. The black points, the decimal numbers, the blue lines, and the green line(s) represent the subgoals, the indexes of subgoals, the distance between source node (r_1) and the other nodes (r_2, r_3), and the distance between the other nodes (r_2 and r_3), respectively. (a) The non-overlapping case. (b) The overlapping case.

To give examples of the proofs, 2D and 3D cases are as follows: As Fig. 4.4(a) shown, the source node is at the intersection point between two solid lines. If the edges of graph are 7 and the subgoals are the vertices of regular heptagon, the solid blue line is greater than the dashed line. It implies that if $|S| > 7$, $H > 1$ for MST; If the number of graph



(a) The center of balls in $z = 0$ (b) The center of balls in $z \geq 0$

Figure 4.6: Illustration of kissing balls of Thm. 4.4

edges is 6 and the subgoals are in the vertices of hexagon, the solid blue line is less than or equal to the dashed line in Fig. 4.4(b). It implies that if $|S| = 7$, $H = 1$ for MST.

Note that if $|S| \leq 10$ in 3D environment, either $\kappa_c = 0$ or $\kappa_c = 1$ is possible. As Fig. 4.2 shows, there are 6 nodes in graph. The height of MST in Fig. 4.2 is 4, i.e., $\kappa_c = 1$.

As Fig. 4.6(a) shows, there are at most six non-overlapping balls in xy plane. By $k(3) = 12$ [44], there are three centers upper xy-plane (see Fig. 4.6(b)). There are at most 10 points (including central point) satisfying $||v_1 - v_j|| \leq ||v_j - v_k||, \forall j \neq k \in \{2, ..., 10\}$.

Theorem 4.5 (The theoretical guarantees of GCB-SPT and GCB-MST)**.** *By Thm. 4.3 and Thm. 4.4, given a submodular monotone set function f, a minimum spanning tree cost function c_{MST} with the condition of subgoals, and a shortest path tree cost function c_{SPT}, the GCB approach is to maximize f subject to the budget B. The performance of a set X with c_{MST} and of a set X with c_{SPT} achieve*

$$f(X) \geq \frac{1}{2}(1 - \frac{1}{e})f(X_{MST}), \tag{4.6}$$

and

$$f(X) \geq \frac{1}{2}(1 - \frac{1}{e})f(X_{SPT}), \tag{4.7}$$

where

$$X_{MST} = arg\max_X\{f(X)|c_{MST}(X) \leq B(\frac{1}{K_{c_{MST}}})\}, \tag{4.8}$$

and

$$X_{SPT} = arg\max_X\{f(X)|c_{SPT}(X) \leq B\}, \tag{4.9}$$

respectively.

Proof. Case1: The theoretical guarantees of GCB-MST is as follows:
By Thm. 4.4, if the subgoal condition is satisfied, $\kappa_c = 1$. Substituting $\kappa_c = 1$ into Eq. 3.5, the budget constraint is $c(X) \leq B(\frac{1}{K_{c_{MST}}})$
$\therefore X_{MST} = arg\max_X\{f(X)|c_{MST}(X) \leq B(\frac{1}{K_{c_{MST}}})\}$
Case2: The theoretical guarantees of GCB-SPT is as follows:
By Thm. 4.3, the κ_c of SPT is 0.
Substituting $\kappa_c = 0$ into Eq. 3.5, the budget constraint is $c(X) \leq B$)
$\therefore X_{SPT} = arg\max_X\{f(X)|c_{SPT}(X) \leq B\}$

□

Corollary 4.5.1 (The theoretical guarantees of GCB-CBST)**.** *Given a submodular monotone set function f, a CBST cost function $\overline{c}$ with $\alpha = 0$ and Eq. 4.3, Eq. 4.4, and a CBST cost function $\breve{c}$ with $\alpha = 1$, the GCB approach is to maximize f subject to the budget B in the case of IPP on the terrain. The performances of a set X achieve*

$$f(X) \geq \frac{1}{2}(1 - \frac{1}{e})f(\overline{X}), \tag{4.10}$$

and

$$f(X) \geq \frac{1}{2}(1 - \frac{1}{e})f(\breve{X}), \tag{4.11}$$

where

$$\overline{X} = arg\max_X\{f(X)|\overline{c}(X) \leq B(\frac{1}{K_{\overline{c}}})\}. \tag{4.12}$$

and

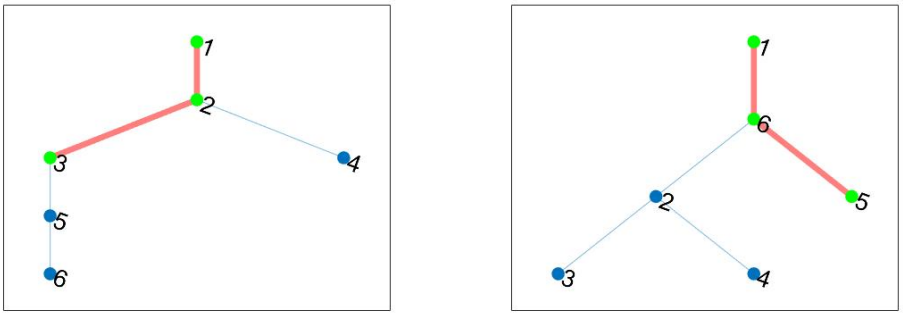
$$\breve{X} = arg\max_X\{f(X)|\breve{c}(X) \leq B\}. \tag{4.13}$$

, respectively.

Proof. The numerator of the CBST objective function is fixed (see Eq. 4.2). When $\alpha = 0$, denominator of the CBST objective function is the same as that of the MST; when $\alpha = 1$, denominator of the CBST objective function is the same as that of the SPT. Hence, the theoretical guarantee is the same as Eq. 4.8 and Eq. 4.9.

□

Although the theoretical guarantees of CBST is the same as MST, the performance of CBST is different from that of MST due to objective function of the tree spanning. The problem is with budget constraints. The CBST spans the tree adopts cost-benefit objective, so the agent will go through the subgoal with more cost-benefit ratio than that of MST. For example, the budget is 3 and the budget constraint is routing in Fig. 1.1(a) , the path in GCB-MST is $\{1, 2, 3\}$ and the path in GCB-CBST is $\{1, 6, 5\}$ in Fig. 4.7(a)(b). The accumulated EPD in GCB-MST is 0.48 and that in GCB-CBST is 0.95.



(a) Path in GCB-MST (b) Path in GCB-CBST

Figure 4.7: Example of difference between GCB-MST and GCB-CBST

5 Proposed algorithms

There are two algorithms: CBST algorithm and terrain monitoring algorithm. The Alg. 1 is to generate the CBST (e.g., from Fig. 1.1 (a) to Fig. 1.1 (b)). The Alg. 2 is to generate the path using GCB algorithm (e.g., from Fig. 1.1 (b) to Fig. 1.1 (c)).

In Alg. 1, the inputs are an undirected graph (G), the source (s). The output is the spanning tree. Line 1 is to build Q (vertices in G). Lines $2-3$ are initializations. Line 5 is to find the distance connecting v_q and v_k, where v_q belongs to the current growing tree (S), and v_k does not belong to the tree current growing tree. Line 6 is to find the path length from the source vertex to v_q along the current growing tree (S). Line 7 is to maximize f/c, where c is the cost function about Prim's algorithm and Dijkstra algorithm. Lines $8-9$ are to update current tree and spanning tree.

Algorithm 1: Cost-benefit spanning tree

Input:
$G = (V, E, w)$: undirected graph
s: the start point
f: objective function
Output: S: spanning tree
Parameters : α (between 0 and 1)

1: $Q = V$ #all of vertices in G
2: $Q = Q \setminus \{s\}$
3: $S = \phi$
4: **while** $Q \neq \phi$ **do**
5: let d_{qk} be the cost edge ;
such that $q \in Q$ and $k \in V \setminus Q$
6: pl_q is distance from source to v_q
7: maximize $(\frac{f}{\alpha \times pl_i + d_{iu}})$ where $i \in Q$, $u \in V \setminus Q$
8: $Q = Q \setminus \{i\}$
9: $S = S \cup (u, i)$
10: **end while**

In Alg. 2, the inputs, S, s, f, c, and B, represent spanning tree built from Alg. 1, the start point, objective function, cost function from

Algorithm 2: Terrain monitoring using CBST

Input:
$S = (V, E, w)$: spanning tree
s: the start point
f: objective function
c: cost function from spanning tree S
B: budget
Output: π: subgoal set

1: $G := \phi$
2: $\pi := \phi$
3: $V^{'} = V$
4: **while** $V^{'} \neq \phi$ **do**
5: **for** $X \in V$ **do**
6: $\Delta_f^X := f(G \cup X) - f(G)$
7: $\Delta_c^X := c(G \cup X) - c(G)$
8: **end for**
9: $X^* = argmax\{\frac{\Delta_f^X}{\Delta_c^X}\}$
10: **if** $c(G \cup X^*) \leq B$ **then**
11: $\pi = \pi \cup X^*$
12: **end if**
13: $V^{'} = V^{'} \setminus X^*$
14: **end while**

spanning tree S using shortest path tree, and a cost budget, respectively. The output is π which is the subgoal set with budget constraint. Lines $1 - 3$ are initializations. Lines $4 - 9$ are to find maximum cost-benefit point in the spanning tree. Lines $10 - 12$ are to pick the point subject to budget constraint. Line 13 is to avoid that the agent picks the point repeatedly.

6 Experiments

To evaluate the performance of different approaches, there are three experiments: the maximization extended probability of detection problem, the search problem, and Tuning α parameters. The EX1 is to evaluate the extended probability of target detection (EPD) of different offline approaches with routing constraints in simulations. The EX2 is to evaluate the search time of different approaches with routing constraints in simulations and real world. The purpose of EX3 is to evaluate the influence of changing α for search results.

6.1 Experiment setup

In the EX1 and EX2, there are two different size maps. The size of map 1 and map 2 are $4 \times 4 \times 3$ (m^3) and $8 \times 8 \times 3$ (m^3), respectively. As shown in Fig. 6.1, the ground set S is generated with $||S|| = 58$. The numbers of subgoals in low altitude is more than that in high altitude, due to the camera field of view (FOV). The index 1 is start point. The experimental setup (e.g., camera and drone parameters) is as shown in Table 6.1.

The cost constraints are 10 and 40, respectively. The cost function c is measured by Euclidean distance. Note that, since computing routing cost is NP-hard problem, c is approximated by $\hat{c}$ via Nearest Neighbor (NN) algorithm [45] which provides a $\frac{3}{2}-$approximation factor.

The target locates in 121 places uniformly. The drone searches 5 times in each place. Hence, there are 605 times. The search processing is as follows: The agent takes off and moves to the next subgoal. If the drone cannot find the target ($P(Z = 1) \leq 95\%$), the agent will go through the next path (point) until finding the target ($P(Z = 1) \geq 95\%$) or utilizing all budgets.

There are two setups in the search experiments. One is offline, the

Table 6.1: Drone hardware setup in simulation.

Parameter	description	value
c_r	range	3.0 (m)
FOV_h	horizontal FOV	60°
FOV_v	vertical FOV	45°
c_{fr}	frame rate	10 (FPS)
h_{max}	flight minimum altitude	0.5 (m)
h_{min}	flight maximum altitude	3.0 (m)
h_{level}	flight altitude levels	0.3 (m)
ω_{max}	limitation of angular velocity	42 (deg/s)
v_{max}	limitation of linear velocity	1 (m/s)

other one is online. The online approach for the drone is to computes the information to decide the next subgoal(s); the offline approach for the drone is to compute all subgoals before taking off. There are two steps in the offline setup. First, the path is generated by algorithm subject to budget constraints offline. Second, the agent flies through the path. There are three steps in the online setup. First, the path (or the point) is generated by algorithm online. Second, the agent flies through the path. Third, the agent repeats first step via new measurements by second step until the terminal condition.

Along with path, the agent updates the probability of target detection (Z) in grid map via Bayesian filter. There are 20×20 and 40×40 grids in map 1 and 2, respectively. When the probability of the cell in grid map is greater than 95% ($P(Z = 1) \geq 95\%$), the agent finds the target. Once the agent finds the target, it will land, and the search time is from taking off to landing. $E[TTD^+]$ denotes the expected time till positive decisions [15].

The map in the real world is a $4 \times 4 \times 2(m^3)$ lobby at the Mathematics department of National Central University (see Fig. 6.2 (a)). The target locates in 5 places randomly. The drone searches 5 times in each place. Hence, there are 125 times. The budget $B = 10(m)$ is adopted in this map. Fig. 6.2 (a) shows the ground set $|S| = 31$ and the fixed start point is the subgoal 1. The target locates in three positions (blue stars) at each search.

As shown in Fig. 6.3, the TD450 is developed by Taiwan Drone

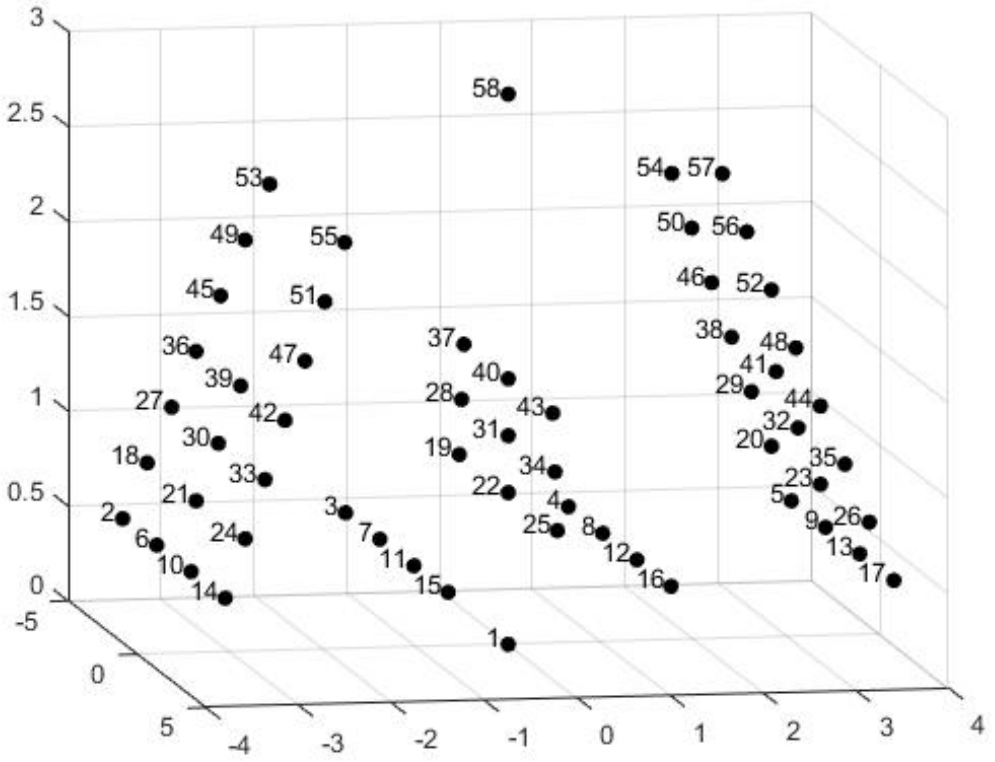


Figure 6.1: Ground set in simulation.

100. The D435i RGBD camera is to explore the environment while T265 camera is to localize itself. The YOLOv5 [46] is adopted to detect objects running on NVIDA Jetson Xavier NX (see Fig. 6.2 (b)). The drone's operating system is Ubuntu 18.04 and runs the Robot Operating System (ROS). The parameters of TD450 hardware setup are shown in Table 6.2.

Table 6.2: Drone hardware setup in real world.

Parameter	description	value
c_r	range	3.0 (m)
FOV_h	horizontal FOV	69°
FOV_v	vertical FOV	42°
c_{fr}	frame rate	30 (FPS)
h_{max}	flight minimum altitude	1.1 (m)
h_{min}	flight maximum altitude	2.0 (m)
h_{level}	flight altitude levels	0.3 (m)
ω_{max}	limitation of angular velocity	120 (deg/s)
v_{max}	limitation of linear velocity	0.2 (m/s)

(a) The lobby environment. (b) The detection outcome of YOLOv5.

Figure 6.2: (a) The black points, black decimal numbers,the index 1 in black represent the subgoals, the subgoal indexes and the source node, respectively. The blue stars and decimal numbers represent the target locations and the subgoal indexes, respectively. (b) The d435i provides the RGB images. The YOLOv5 detects the target in the image. The blue box represents the target location in the image with the detection probability.

6.2 EX1: EPD maximization

In the experiment, the GCB-CBST is compared with four methods: A-optimality with CMA-ES [2], GCB [5], GCB-MST [10], and GCB-

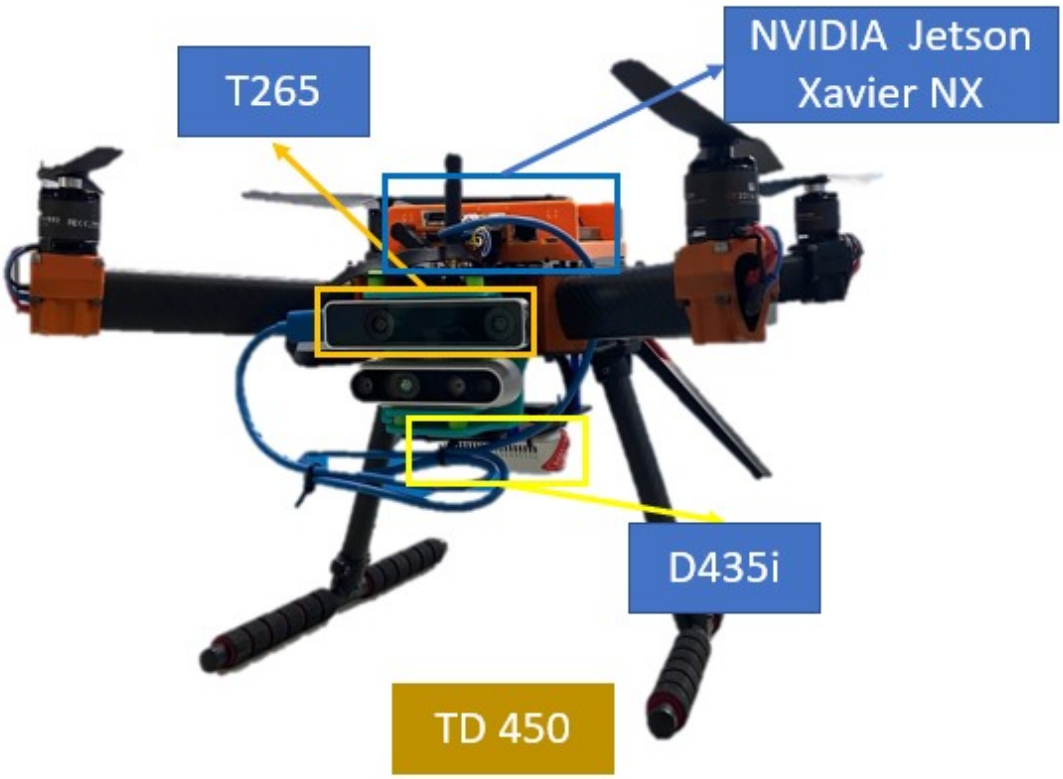


Figure 6.3: The TD450, sensors and development board.

SPT. CMA-ES is online method, the others are offline methods.

The experiment results in the Gazebo simulation are summarized in Table 6.3. Since CMA-ES is online algorithm, the accumulated EPD is not available. GCB-CBST outperforms other approaches in the accumulated EPD in map 1 and 2. The tree structures of MST, SPT, and CBST in map 1 and 2 are shown in Fig. 6.4 and 6.5, respectively. As shown in Fig. 6.4 (b) and 6.5 (b), the tree structure of SPT are inefficient seeing due to the repeated paths. Since the path in GCB-SPT have high cost-benefit, the accumulated EPD of GCB-SPT is higher than GCB-MST. These experiments show that GCB-CBST outperforms the other three methods in the maximal EPD problem.

Table 6.3: EPD maximization experiment results. The accumulated EPD (AEPD) as follows.

Method	AEPD (map1)	AEPD (map2)
A-opt with CMA-ES	NA	NA
GCB	0.90	0.80
GCB-CBST	**0.99**	**0.84**
GCB-MST	0.489	0.47
GCB-SPT	0.843	0.52

6.3 EX2: Search experiment

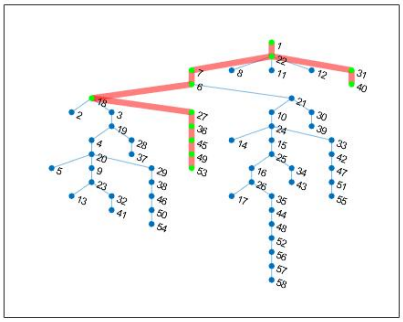
The goal of the drone is to find the target as soon as possible subject to budget constraint, so the metrics in this experiment are expected time till positive decisions ($\mathbb{E}[\text{TTD}^+]$) and success rate.

The parameters are same as EX1.

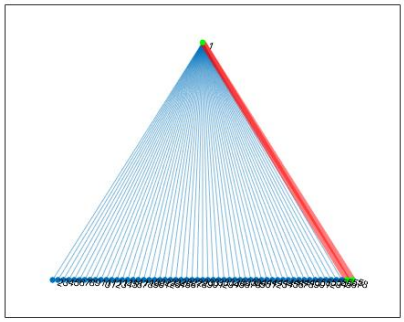
6.3.1 Simulation

To evaluate the search performance, the expected time till positive decision ($\mathbb{E}[\text{TTD}^+]$) is compared (see Fig. 6.6 and 6.7). GCB-CBST outperforms other four methods in $\mathbb{E}[\text{TTD}^+]$. GCB-CBST and GCB-SPT are competitive in map 1 since the drone moves to high cost-benefit subgoals. However, the success rate in GCB-SPT is less than that in GCB-CBST since the path in GCB-SPT is inefficient.

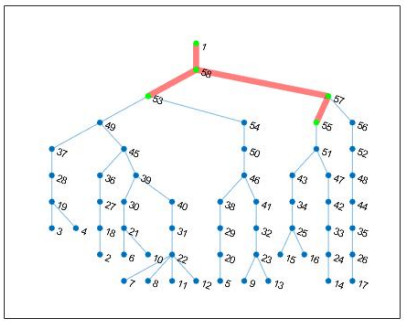
The success rates of the GCB and GCB-CBST are competitive in map 2 (see Table 6.4). The success rate of the GCB is higher than that of the GCB-CBST in map 1 (see Table 6.4). In the GCB-CBST



(a) The tree structure of MST

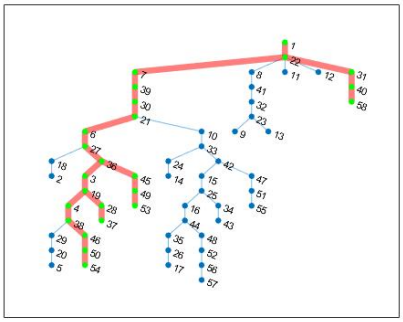


(b) The tree structure of SPT

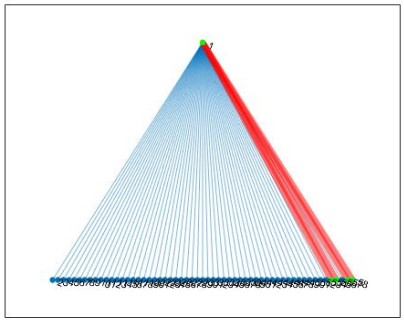


(c) The tree structure of CBST

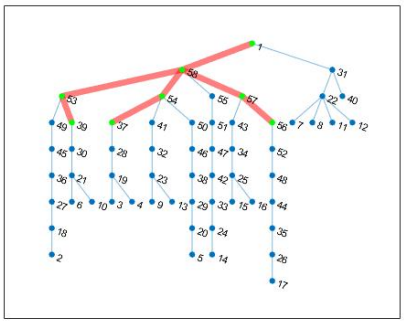
Figure 6.4: The blue circles represent the ground set, the green circle represent the picked subgoals, and the red lines represent the path, respectively. There are three tree structures in map1.



(a) The tree structure of MST



(b) The tree structure of SPT



(c) The tree structure of CBST

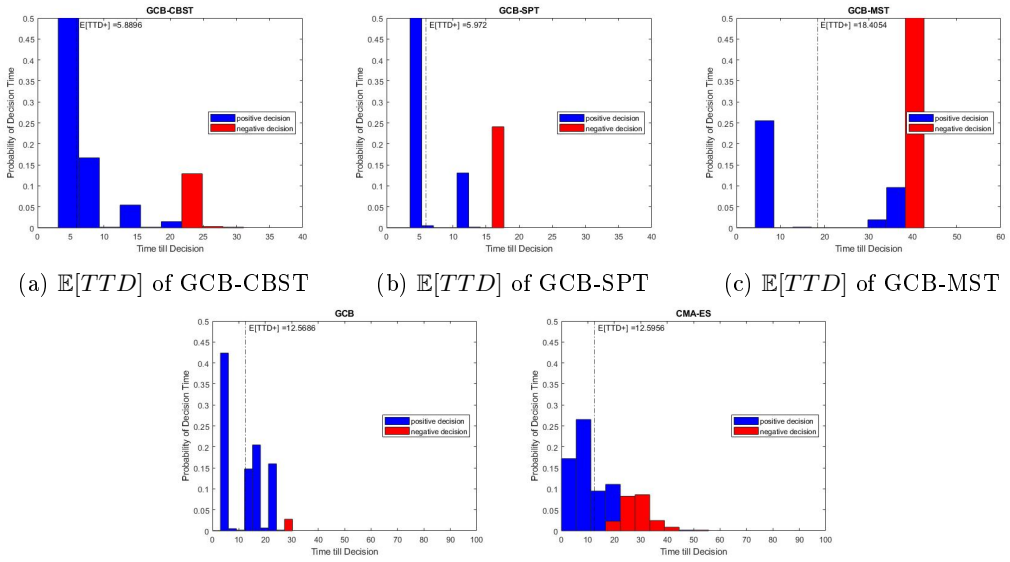
Figure 6.5: The blue circles represent the ground set, the green circle represent the picked subgoals, and the red lines represent the path, respectively. There are three tree structures in map2.

path, the drone moves to the repeated subgoals. Hence, it has fewer false detections than GCB does. GCB-CBST attains the a little less success rate than GCB in map1 because the drone moves to the repeated subgoals in GCB-CBST.

In summary, GCB-CBST is the best method for search tasks, since it can move through the high cost-benefit subgoals immediately and repeatedly.

Table 6.4: Search experiment results reveal indicators success rate and $\mathbb{E}[\text{TTD}^+]$ in map1 and map2.

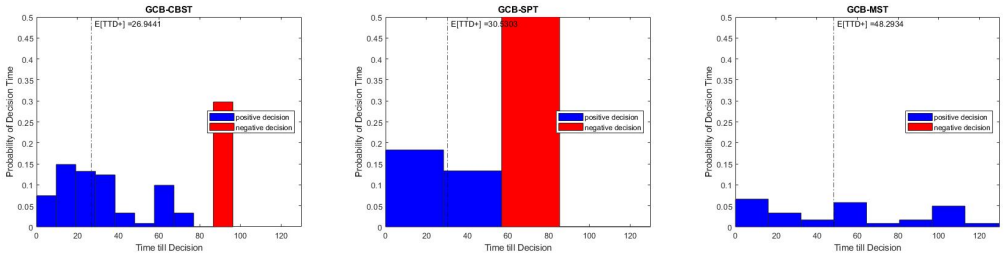
Method	Map1		Map2	
	$\mathbb{E}[\text{TTD}^+]$	success rate	$\mathbb{E}[\text{TTD}^+]$	success rate
CMA-ES	12.6	0.77	85.7	0.28
GCB	12.6	**0.96**	30.4	**0.73**
GCB-CBST	**5.9**	0.8	**26.9**	0.705
GCB-MST	18.4	0.44	48.3	0.30
GCB-SPT	6.0	0.75	30.5	0.43



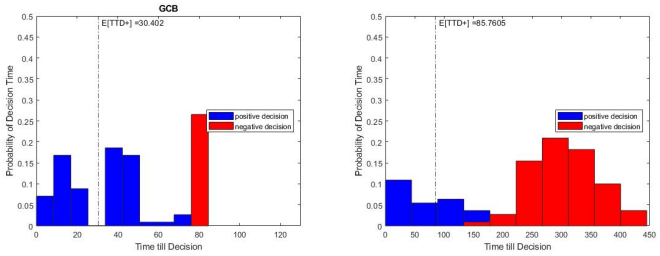
(a) $\mathbb{E}[TTD]$ of GCB-CBST (b) $\mathbb{E}[TTD]$ of GCB-SPT (c) $\mathbb{E}[TTD]$ of GCB-MST

(d) $\mathbb{E}[TTD]$ of GCB (e) $\mathbb{E}[TTD]$ of CMA-ES

Figure 6.6: $\mathbb{E}[TTD]$ of different tree-structured cost in map1. The x-axis and y-axis represent time (s) and probability, respectively.



(a) $\mathbb{E}[TTD]$ of GCB-CBST (b) $\mathbb{E}[TTD]$ of GCB-SPT (c) $\mathbb{E}[TTD]$ of GCB-MST



(d) $\mathbb{E}[TTD]$ of GCB (e) $\mathbb{E}[TTD]$ of CMA-ES

Figure 6.7: $\mathbb{E}[TTD]$ of different tree-structured cost in map2. The x-axis and y-axis represent time (s) and probability, respectively.

6.3.2 Real world

The tree structures of GCB-CBST, GCB-MST, and GCB-SPT are shown in Fig. 6.8. GCB-CBST outperforms the other four methods in $\mathbb{E}[\text{TTD}^+]$ (see Table 6.5). The height in real world is lower than that in the simulation, so the path in GCB-SPT does not have high coverage. The $\mathbb{E}[\text{TTD}^+]$ in GCB-SPT is the worst.

The GCB-CBST outperforms the other four methods in success rate (see Table 6.5), since the GCB-CBST must go to subgoals via tree structure, and the drone moves to the repeated subgoals which have high cost-benefit values. The GCB-MST and GCB-SPT are the worst in success rate.

These experiments show that GCB-CBST outperforms the other four benchmarks in search problems.

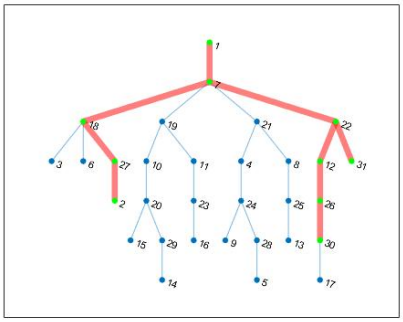
Table 6.5: Search experiment results reveal indicators success rate and $\mathbb{E}[\text{TTD}^+]$ in real world.

	$\mathbb{E}[\text{TTD}^+]$	success rate
CMA-ES	86.91	0.88
GCB	69.23	0.88
GCB-CBST	**51.92**	**1.00**
GCB-MST	68.77	0.52
GCB-SPT	79.59	0.68

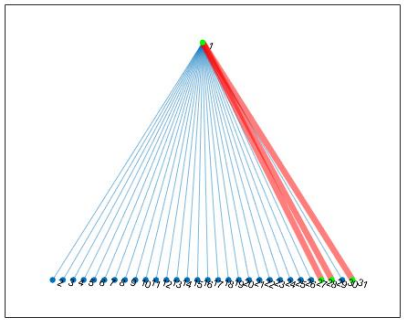
6.4 EX3: α tuning

The α ranges from 0 to 1 with an interval of 0.05. In the map1, the alpha values generate 4 kinds of tree structures. the alpha ranges are $0 \sim 0.15$, $0.2 \sim 0.25$, $0.3 \sim 0.4$ and $0.4 \sim 1$. The corresponding sets are S_0, S_1, S_2 and S_3 in Fig. 6.9. In the map2, the alpha values generate 6 kinds of tree structures. the alpha ranges are 0, 0.05, $0.1 \sim 0.15$, 0.2, $0.25 \sim 0.5$ and $0.55 \sim 1$. The corresponding sets are M_0, M_1, M_2, M_3, M_4 and M_5 in Fig. 6.10.

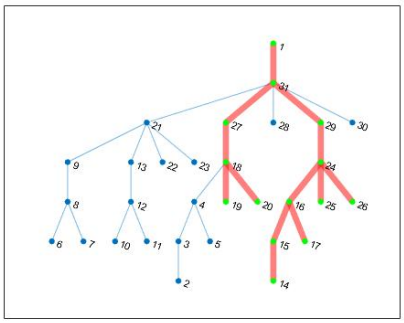
As shown in Table 6.6, since map 1 is not big enough, the results in different tree structures in GCB-CBST show insignificant differences. As shown in Table 6.7, there are significant difference between $\alpha = 0$ and $\alpha = 1$ since the map 2 $(8 \times 8 \times 3(m^3))$ is bigger than map 1 $(4 \times 4 \times 3(m^3))$. The cost function with parameter $\alpha = 0$ is the best



(a) The tree structure of MST



(b) The tree structure of SPT



(c) The tree structure of CBST

Figure 6.8: Three tree structures in the lobby environment. The blue circles and green circles represent the ground set and the picked subgoals, respectively. The red lines represent the paths.

in $\mathbb{E}[\text{TTD}^+]$, because the tree structure is minimum weight spanning tree. In the MST algorithm, it is possible to generate edges with the minimum total weight, which results in shorter distances between subgoals. Consequently, a better $\mathbb{E}[\text{TTD}^+]$ can be achieved. In the map 2, the M_3, M_4 and M_5 are inefficient compared to M_0, M_1 and M_2 in Fig. 6.10.

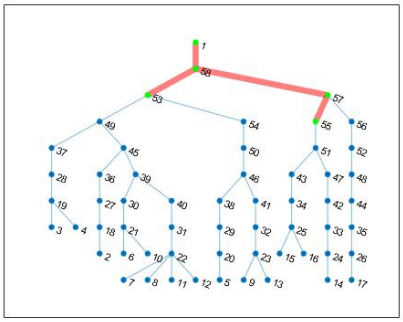
In summary, the performance when α approaches 0 is better than the performance when α approaches 1 due to the path efficiency. How to tune an optimal α is still an issue. The $\alpha = 0$ could be a fast solution in search experiments.

Table 6.6: Search experiment results reveal indicators success rate and $\mathbb{E}[\text{TTD}^+]$ in map 1 with different α.

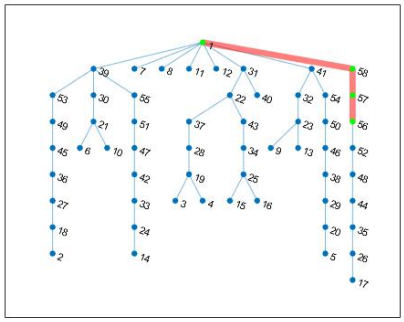
α	E[TTD$^+$]	Success rate
$0 \sim 0.15$	7.85	**0.74**
$0.2 \sim 0.25$	**5.76**	0.60
$0.3 \sim 0.4$	6.48	0.63
$0.45 \sim 1$	6.58	0.64

Table 6.7: Search experiment results reveal indicators success rate and $\mathbb{E}[\text{TTD}^+]$ in map 2 with different α.

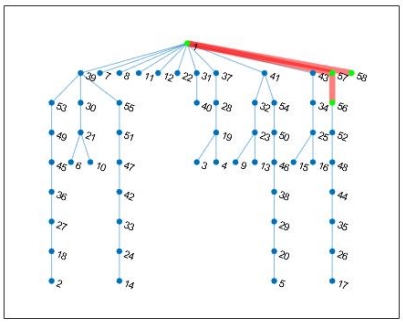
α	E[TTD$^+$]	Success rate
0	**26.9**	0.70
0.05	38.8	**0.83**
$0.1 \sim 0.15$	38.5	**0.83**
0.2	34.1	0.62
$0.25 \sim 0.5$	38	0.50
$0.55 \sim 1$	30.5	0.43



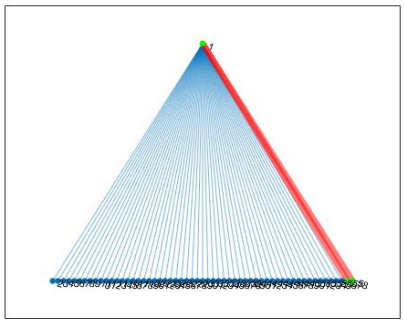
(a) The tree structure of S_0



(b) The tree structure of S_1

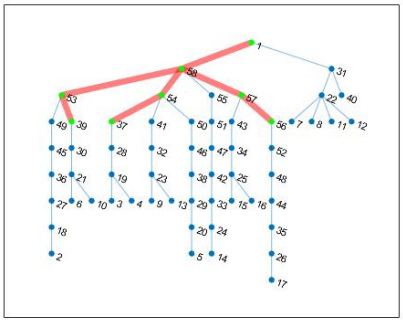


(c) The tree structure of S_2

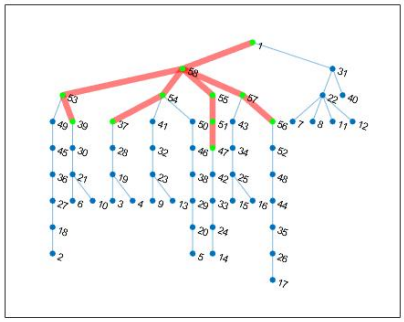


(d) The tree structure of S_3

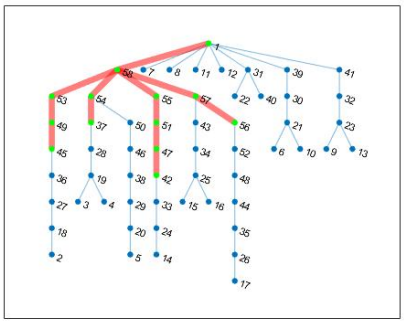
Figure 6.9: The blue circles represent the ground set, the green circle represent the picked subgoals, and the red lines represent the path, respectively. There are three tree structures in map 1.



(a) The tree structure of M_0

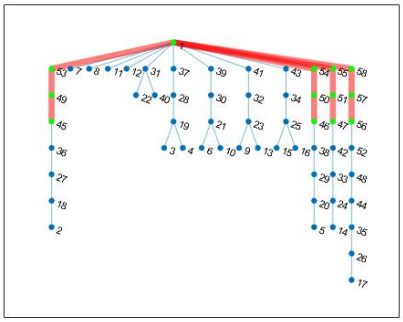


(b) The tree structure of M_1

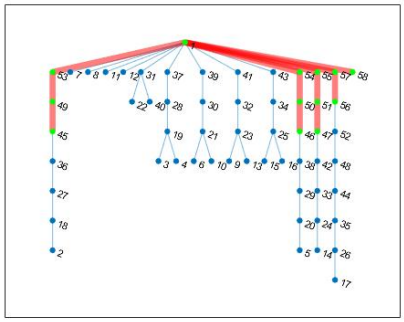


(c) The tree structure of M_2

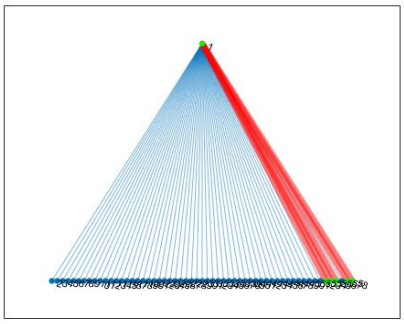
Figure 6.10: The blue circles represent the ground set, the green circle represent the picked subgoals, and the red lines represent the path, respectively. There are six tree structures in map 2.



(d) The tree structure of M_3



(e) The tree structure of M_4



(f) The tree structure of M_5

Figure 6.10: The blue circles represent the ground set, the green circle represent the picked subgoals, and the red lines represent the path, respectively. There are six tree structures in map 2.

7 Conclusions and future work

This research proposes the CBST tree structure to solve submodular maximization problems subject to budget constraint. The research is summarized as follows: First, to the best of our knowledge, this is the first work to span cost-benefit tree outperforming benchmark methods. Second, due to the κ_c of the different tree structure, the theorems show the theoretical guarantees in the different tree structures. Third, the experiments show that the proposed CBST using GCB algorithm outperforms the benchmark approaches.

The future work of this research is as follows: First, The CBST using GCB algorithm outperforms the benchmark in search experiments in the empty map. If there are obstacles in the map, the CBST spanning tree could not have better performance than the other methods. Second, the GCB with tree structured graph cost function is offline approach. How to apply adaptive submodularity to tree structures is another research direction. Finally, the proposed algorithm is applied to a static target. How to extend the proposed algorithm to multiple targets and moving targets is another potential research.

References

[1] Marija Popović, Teresa Vidal-Calleja, Gregory Hitz, Inkyu Sa, Roland Siegwart, and Juan Nieto. Multiresolution mapping and informative path planning for uav-based terrain monitoring. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1382–1388, 2017.

[2] Marija Popović, Teresa Vidal-Calleja, Gregory Hitz, Jen Jen Chung, Inkyu Sa, Roland Siegwart, and Juan Nieto. An informative path planning framework for uav-based terrain monitoring. *Autonomous Robots*, 44(6):889–911, 2020.

[3] 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.

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

[5] Haifeng Zhang and Yevgeniy Vorobeychik. Submodular optimization with routing constraints. *Proceedings of the AAAI conference on artificial intelligence*, 30(1), 2016.

[6] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[7] Merrill M Flood. The traveling-salesman problem. *Operations research*, 4(1):61–75, 1956.

[8] Tal Grossman and Avishai Wool. Computational experience with approximation algorithms for the set covering problem. *European journal of operational research*, 101(1):81–92, 1997.

[9] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973.

[10] 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*, 8(4):1927–1934, 2023.

[11] Lawrence D Stone. *Theory of optimal search*. Elsevier, 1976.

[12] Frederic Bourgault, Tomonari Furukawa, and Hugh F Durrant-Whyte. Coordinated decentralized search for a lost target in a bayesian world. *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1:48–53, 2003.

[13] KE Trummel and JR Weisinger. The complexity of the optimal searcher path problem. *Operations Research*, 34(2):324–327, 1986.

[14] Timothy H Chung and Joel W Burdick. A decision-making framework for control strategies in probabilistic search. *Proceedings IEEE International Conference on Robotics and Automation*, pages 4386–4393, 2007.

[15] Timothy H Chung and Joel W Burdick. Analysis of search decision making using probabilistic search strategies. *IEEE Transactions on Robotics*, 28(1):132–144, 2011.

[16] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[17] Kuo-Shih Tseng and Bérénice Mettler. Near-optimal probabilistic search via submodularity and sparse regression. *Autonomous Robots*, 41(1):205–229, 2017.

[18] Haye Lau, Shoudong Huang, and Gamini Dissanayake. Discounted mean bound for the optimal searcher path problem with non-uniform travel times. *European journal of operational research*, 190(2):383–397, 2008.

[19] Eva Besada-Portas, Luis de la Torre, Jesus M de la Cruz, and Bonifacio de Andrés-Toro. Evolutionary trajectory planner for multiple uavs in realistic scenarios. *IEEE Transactions on Robotics*, 26(4):619–634, 2010.

[20] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.

[21] Marina Torres, David A Pelta, José L Verdegay, and Juan C Torres. Coverage path planning with unmanned aerial vehicles for 3d terrain reconstruction. *Expert Systems with Applications*, 55:441–451, 2016.

[22] Chao Qian, Jing-Cheng Shi, Yang Yu, and Ke Tang. On subset selection with general cost constraints. *IJCAI*, 17:2613–2619, 2017.

[23] Chao Bian, Chao Feng, Chao Qian, and Yang Yu. An efficient evolutionary algorithm for subset selection with general cost constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3267–3274, 2020.

[24] Ioannis K Nikolos, Kimon P Valavanis, Nikos C Tsourveloudis, and Anargyros N Kostaras. Evolutionary algorithm based offline/online path planner for uav navigation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(6):898–912, 2003.

[25] Gregory Hitz, Enric Galceran, Marie-Ève Garneau, François Pomerleau, and Roland Siegwart. Adaptive continuous-space informative path planning for online environmental monitoring. *Journal of Field Robotics*, 34(8):1427–1449, 2017.

[26] Shervin Javdani, Matthew Klingensmith, J Andrew Bagnell, Nancy S Pollard, and Siddhartha S Srinivasa. Efficient touch based localization through submodularity. *IEEE International Conference on Robotics and Automation*, pages 1828–1835, 2013.

[27] Amarjeet Singh. Nonmyopic adaptive informative path planning for multiple robots. 2009.

[28] Joseph B Kadane and Herbert A Simon. Optimal strategies for a class of constrained sequential problems. *The Annals of Statistics*, 5(2):237–255, 1977.

[29] Marija Popovic, Gregory Hitz, Juan Nieto, Inkyu Sa, Roland Siegwart, and Enric Galceran. Online informative path planning for active classification using uavs. *arXiv preprint arXiv:1609.08446*, 2016.

[30] Brian Yamauchi. Frontier-based exploration using multiple robots. *Proceedings of the second international conference on Autonomous agents*, pages 47–53, 1998.

[31] Nikolaus Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation*, pages 75–102, 2006.

[32] Micah Corah and Nathan Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice. *Autonomous Robots*, 43:485–501, 2019.

[33] 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.

[34] Geoffrey Hollinger and Sanjiv Singh. Proofs and experiments in scalable, near-optimal search by multiple robots. *Proceedings of Robotics: Science and Systems*, 1, 2008.

[35] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. *Proceedings of the 5th international conference on Information processing in sensor networks*, pages 2–10, 2006.

[36] Peter Stobbe and Andreas Krause. Learning fourier sparse set functions. *Artificial Intelligence and Statistics*, pages 1125–1133, 2012.

[37] Robert Clay Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957.

[38] EW Dijkstra. A note on two problems in connexion with graphs. *Numerische Math.*, 1:269–271, 1959.

[39] Charles J Alpert, TC Hu, Jen-Hsin Huang, and Andrew B Kahng. A direct combination of the prim and dijkstra constructions for improved performance-driven global routing. *IEEE International Symposium on Circuits and Systems*, pages 1869–1872, 1993.

[40] Charles J Alpert, Wing-Kai Chow, Kwangsoo Han, Andrew B Kahng, Zhuo Li, Derong Liu, and Sriram Venkatesh. Prim-dijkstra revisited: Achieving superior timing-driven routing trees. *Proceedings of the International Symposium on Physical Design*, pages 10–17, 2018.

[41] Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3):251–274, 1984.

[42] Kuo-Shih Tseng. *Learning in human and robot search: Subgoal, submodularity, and sparsity.* PhD thesis, University of Minnesota, 2016.

[43] Rei-Shu Shieh. Map explorations via dynamic tree-structured graph. Master's thesis, National Central University, 2023.

[44] Kurt Schütte and Bartel Leendert van der Waerden. Das problem der dreizehn kugeln. *Mathematische Annalen*, 125(1):325–334, 1952.

[45] Daniel J Rosenkrantz, Richard E Stearns, and Philip M Lewis, II. An analysis of several heuristics for the traveling salesman problem. *SIAM journal on computing*, 6(3):563–581, 1977.

[46] 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.