MSTC*:Multi-robot Coverage Path Planning under Physical Constraints

Jingtao Tang, Chun Sun and Xinyu Zhang

Abstract— **For large-scale tasks, coverage path planning (CPP) can benefit greatly from multiple robots. In this paper, we present an efficient algorithm MSTC* for multi-robot coverage path planning (mCPP) based on spiral spanning tree coverage (Spiral-STC). Our algorithm incorporates strict physical constraints like terrain traversability and material load capacity. We compare our algorithm against the state-of-the-art in mCPP for regular grid maps and real field terrains in simulation environments. The experimental results show that our method significantly outperforms existing spiral-STC based mCPP methods. Our algorithm can find a set of well-balanced workload distributions for all robots and therefore, achieve the overall minimum time to complete the coverage.**

I. INTRODUCTION

Coverage path planning (CPP) is the problem of determining a set of paths that cover the area of interest while avoiding obstacles[1]. Coverage path planning has many indoor and outdoor robotic applications, such as vacuum cleaning robots[2], autonomous underwater vehicles[3], unmanned aerial vehicles[4], demining robots[5], automated harvesters[6], planetary exploration[7], search and rescue operations[8], lawn mowers[9], massive afforestation[10].

Coverage path planning has been received a lot of attention in robotics and there are a considerable research work addressing this problem[1]. This includes cellular decomposition[11], [12], [13], gird map[14], spanning tree coverage[15], neural network-based coverage[16], [17], graph-based coverage[18], optimal coverage[19], [20], coverage under uncertainty[21], [22]. Most these approaches were designed mainly for a single robot.

For large-scale tasks, coverage path planning can benefit greatly from multi-robot systems. First, a multi-robot system clearly completes the task fast due to workload distribution. Second, multiple robots can collaborate with each other to accomplish complex tasks efficiently. Third, multi-robots improve robustness in case of failure of some robots. Though there are many advantages using multiple robots, the research in multi-robot coverage path planning is relatively limited since some extra factors (e.g., data sharing, complex path generation, task division/allocation, physical constraints, etc.) need to be taken into account. Many approaches extended single-robot algorithms to handle multi-robot systems using workload division/distribution[23].

Despite the exciting potential applications, designing a scalable and practical multi-robot coverage path planning

The authors are with Shanghai Key Laboratory of Trustworthy Computing, Engineering Research Center of Software/Hardware Co-design Technology and Application (MoE) and School of Software Engineering, East China Normal University, Shanghai. Xinyu Zhang is the corresponding author. E-mail: {xyzhang}@sei.ecnu.edu.cn.

(mCPP) algorithm under strict physical constraints remains a challenging problem. Our research was developed from an ambitious program for tree planting robots to restore vast degraded lands. These terrains may exhibit complex surface and topology. The robots have limited energy and workload/material capacity (e.g., 100 tree saplings per load for planting robots and 500kg water per load for watering robots). Coverage path planning with physical constraints is a relatively new topic and energy constraints were often considered in limited research literatures[24], [25], [26], [27].

Main Results: We propose a novel method namely MSTC* (Multi-robot Spanning Tree Coverage Star), to solve the mCPP problem under physical constraints of traversability and limited workload/material capacity. We treat mCPP as the problem of partitioning a topological loop and assign each partition to one robot. To find a set of well-balanced partitions, we start with a set of naïve partitions and iteratively generate balanced partitions for all robots by minimizing the maximum weights. Our balanced-MSTC* uses the strategy of balanced cut to search the most unbalanced two partitions (with the maximum and minimum weights, respectively). This strategy is a greedy algorithm and is able to gradually approximate the optimal partitions. Our algorithm can find a set of well-balanced workload distributions for all robots and therefore, achieve the overall optimal time to complete the coverage. We compare our algorithm against other spiral-STC based mCPP methods on regular grid maps and real field terrains in simulation environments. The results show that our MSTC* algorithm outperforms the state-of-the-art, like classic MSTC (MSTC-NB)[28], MSTC with backtracking (MSTC-BO)[28] and Multi-robot Forest Coverage (MFC)[29].

II. RELATED WORK

Coverage path planning is well studied for a single robot and we refer readers to [14], [1], [23] for extensive survey. Here, we briefly review the literature relevant to our work.

Our method is inspired by spiral spanning tree coverage (Spiral-STC) [15] for a single robot and multi-robot spanning tree coverage (MSTC) [28] for unweighted graph. The latter improved the efficiency and robustness by introducing redundancy and backtracking optimization using multiple robots. Agmon et al. constructed a spanning tree by minimizing the time to complete the coverage [30], [31]. Zheng et.al. proposed multi-robot forest coverage (MFC) to solve the mCPP problem using multiple minimal spanning trees to cover the terrain generated by min-max tree cover algorithm [32]. The algorithm works for both unweighted terrains [29] and weighted terrains [33], [34]. Most algorithms assumed that

robots can fully cover the environment without recharging or refilling. However, in the real-world applications, many physical constraints need to be considered. Coverage path planning with physical constraints is a relatively new topic. In some recent work[24], [25], [26], [27], energy limitations were considered on coverage path planning for a single robot. Moreover, Sipahioglu et al. proposed a generalized Voronoi diagram based method to solve the problem of mCPP under energy capacity[35]. Huang et al. developed a quadtree and spiral-STC based method to adapt mCPP to different land types[36]. Our algorithm incorporates strict physical constraints like terrain traversability and material load capacity.

III. PROBLEM DEFINITION & PRELIMINARIES

A. Problem Definition

The goal of mCPP is to cover a given terrain using multiple robots. The terrain is divided into a large number of cells and the cell dimensions depend on specific applications. For example, in our program for tree planting, the spacing between the lines in plantation is 5 meters and the spacing of plants within a line is 3 meters. Then we represent the terrain as a graph, namely *covering graph*, denoted by $\mathcal{G}$ (see Fig. 1-(a)). The nodes in $\mathcal{G}$ is covering nodes, denoted by π. Two adjacent nodes are connected by an edge e. In spiral-STC based algorithms, a spanning graph is used to efficiently generate coverage paths. Here, we denote a spanning graph by $\mathcal{H}$ and its nodes are *spanning nodes*. Note that a covering node is associated to only one spanning node. Both $\mathcal{G}$ and $\mathcal{H}$ are edge-weighted graphs. The edge weights $\|e\|$ are shown in Fig. 1.

(a) Covering Graph $\mathcal{G}$ & Spanning Graph $\mathcal{H}$　(b) $\|e\|$ in $\mathcal{G}$　(c) $\|e\|$ in $\mathcal{H}$
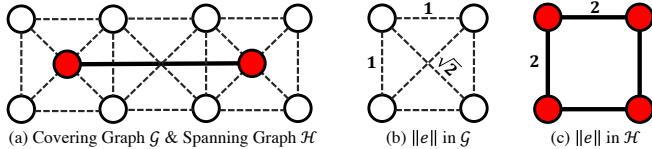
Fig. 1. Covering graph $\mathcal{G}$ is represented by white nodes and dotted edges, and spanning graph $\mathcal{H}$ is represented by red nodes and solid edges. (a) A spanning node is generated from four adjacent covering nodes; (b) A weight (1 or $\sqrt{2}$) is assigned to the edges of $\mathcal{G}$; (c) A weight 2 is assigned to the edges of $\mathcal{H}$.

Given $\mathcal{G}$ and k robots, a coverage path Π_i travelled by robot $\mathcal{R}_i$ is a set of nodes $\{\pi_i^j\}$ in $\mathcal{G}$ (see Fig. 2). For robot $\mathcal{R}_i$, its accumulating weight (cost) is denoted by $\mathcal{W}_{\Pi_i}$. We aim at computing a set of coverage paths $\{\Pi_1, \Pi_2, \ldots, \Pi_k\}$ for k robots through minimizing the maximum of $\mathcal{W}_{\Pi_i}$. Then the problem can be formulated as follows.

$$\underset{\{\Pi_i\}}{\arg\min} \left(\max_{1 \le i \le k} (\mathcal{W}_{\Pi_i}) \right) \tag{1}$$

Besides requirements described in [1], the mCPP problem in this paper has the following constraints:

- *Depots*: a robot $\mathcal{R}_i$ has its own depot, denoted by π_i^d.
- *Cover and return*: a robot returns to its individual depot when its tasks are accomplished.
- *Workload capacity*: a robot has limited material capacity per load and need to immediately return to its depot to refill when its material runs out.
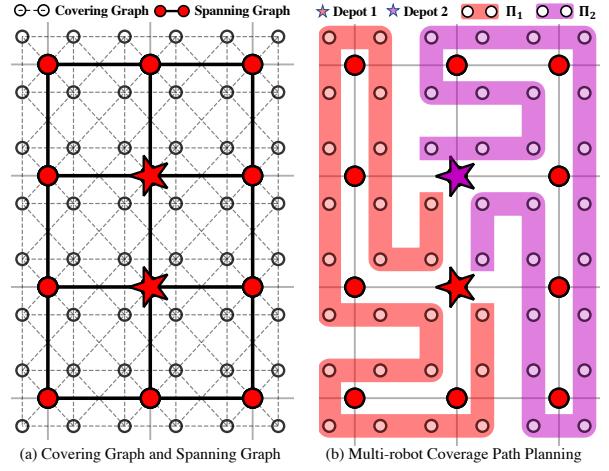

(a) Covering Graph and Spanning Graph　(b) Multi-robot Coverage Path Planning

Fig. 2. Multi-robot coverage path planning using our method MSTC*. (a) Covering graph $\mathcal{G}$ and spanning graph $\mathcal{H}$ constructed from a given terrain; (b) Coverage planning results using two robots starting from their individual depots (highlighting with stars). Their coverage paths are highlighted in red and purple, respectively. Note that, a spanning graph is used to efficiently generate coverage paths.

B. Spiral-STC for A Single Robot

In this section, we introduce the spiral-STC algorithm that inspires our work. Spiral-STC was originally proposed for a single robot. Given a covering graph $\mathcal{G}$ and its associating spanning tree $\mathcal{H}$, spiral-STC performs a counter-clockwise depth-first-search in $\mathcal{H}$ and then generates a circumnavigating coverage path in $\mathcal{G}$ by following the right-side of traversal route (i.e., ordered spanning tree edges) in $\mathcal{H}$.

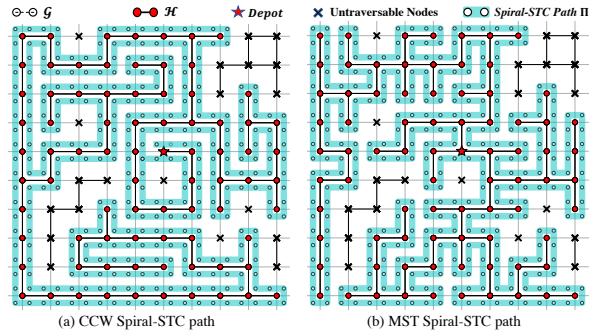(a) CCW Spiral-STC path　(b) MST Spiral-STC path

Fig. 3. Coverage path generated using Spiral-STC in counter-clockwise manner (left) and further improved using MST (right).

The original Spiral-STC algorithm does not consider the impact of edge weights on the final coverage cost $\mathcal{W}$. Minimal-Spanning-Tree (MST) generated a minimal-weighted coverage path on $\mathcal{H}$ and $\mathcal{G}$ by dropping off the edges with high weights. Fig. 3 gives an example to show coverage planning on a 10×10 regular grid graph using standard Spiral-STC and an improvement by MST. Their accumulating costs are $\mathcal{W} = 227.02$ and $\mathcal{W} = 207.20$, respectively.

IV. TERRAIN TRAVERSABILITY MAP

A given terrain needs to be processed to obtain traversability map and then generate covering graph, in which the robots

can freely move without worrying clear obstacles. Here, we use both digital elevation model (DEM) [37] and satellite map [38] to perform reliable traversability assessment. The DEM is often obtained by the aerial photogrammetric reconstruction. The use of DEM allows us to perform an analysis of the geometric properties of the given terrain, like terrain slope and height continuity/discontinuity. Fig. 4 shows an example of DEM data and satellite map.

In our traversability assessment, three main aspects considered herein are

- Steep regions are unreachable or have the high risk of turnover and sliding for ground mobile robots. Therefore, a slope threshold $25°$ is specified based on our preliminary experiments.
- Isolated regions must be removed from traversability map since they are unreachable from any robot depot.
- Non-working regions must not be included in traversability map. These regions include forest, shrubland, marsh, lake, etc.
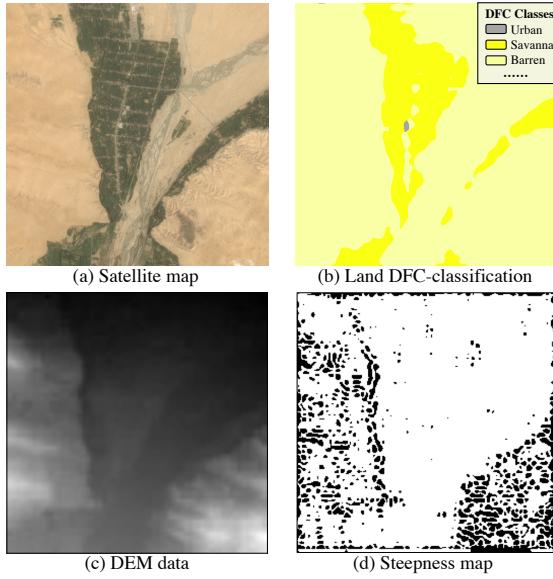

(a) Satellite map

(b) Land DFC-classification

(c) DEM data

(d) Steepness map

Fig. 4. Traversability map for a terrain at location (77.88°E, 37.35°N). (a) The original satellite map; (b) Land DFC-classification (only non-plantation regions are displayed); (c) The DEM data representing height information; (d) Steepness traversability map (steepness threshold is set to 25°). Traversable/non-traversable regions resulted from steepness analysis, are highlighted in white and black, respectively.

Given a terrain and its initial map (graph), we examine all the edges and remove those edges with the slope greater than a specified threshold (e.g. 25°). The goal is to avoid steep paths as well as obstacles. Then, we remove all unconnected sub-graphs or nodes from the initial graph. As a result, we obtain a traversability map including reachable nodes and their traversal costs. For a given edge e, its cost is defined by the normalized slope $\hat{\theta}_e$, and the edge weight contributed by distance $\|e\|$. That is,

$$w_e = \alpha \cdot \|e\| + \beta \cdot \hat{\theta}_e, \quad e \in \mathcal{G} \text{ or } \mathcal{H}, \quad (2)$$

where α and β are the coefficients distinguishing distance contribution and slope contribution. These two parameters can be tuned with the consideration of the significance of distance traversal and the risk of sliding along slopes. Moreover, $\hat{\theta}_e$ is defined as

$$\hat{\theta}_e = \frac{\theta_e - \theta_{min}}{\theta_{max} - \theta_{min}}, \quad (3)$$

where θ_{max} and θ_{min} is the maximum and minimum slope. Note that θ_{max} is the slope threshold (e.g. 25°) used in steepness analysis.

To classify workable and non-workable regions in the satellite map, we use a DNN-based pixel-wise imagery segmentation technique. More specifically, we use the DeepLab Neural Network structure suggested in [39], [40] and apply it to the SEN12M and DFC2020 datasets [41], [42].

An example is given in Fig. 4. The original satellite map is given in Fig. 4-(a). The traversable regions are shown in Fig. 4-(b). The DEM data is given Fig. 4-(c). Its steepness analysis and the results are given in Fig. 4-(d).

With the consideration of three aspects mentioned above, the two traversability maps are merged into one. Then covering graph $\mathcal{G}$ and spanning graph $\mathcal{H}$ are constructed for these maps.

V. OUR ALGORITHM

A. Coverage Path Partition

The coverage path planning for multiple robots can be treated as the problem of partitioning a topological loop (see Fig. 5). Inspired by MSTC, our algorithm MSTC* aims at partitioning the entire coverage path Π into k partitions and assign each partition to one robot.
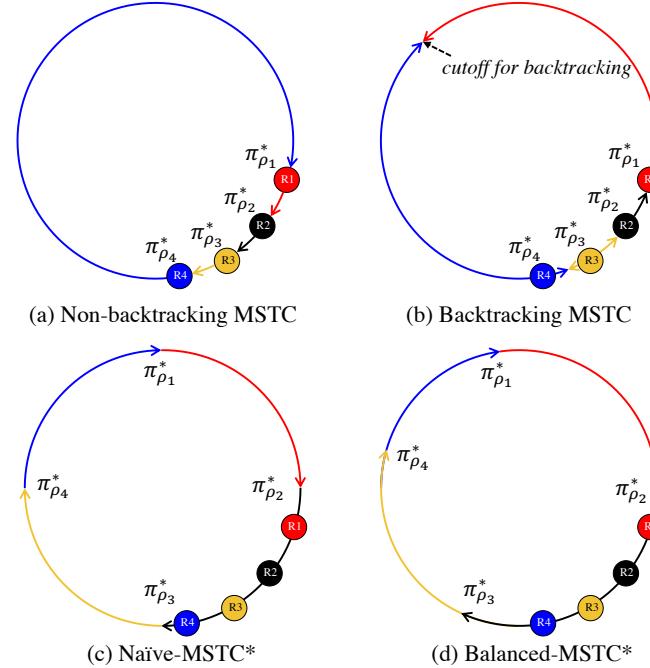

(a) Non-backtracking MSTC

(b) Backtracking MSTC

(c) Naïve-MSTC*

(d) Balanced-MSTC*

Fig. 5. Coverage path partition strategies. (a) MSTC; (b) Backtracking MSTC; (c) Naïve-MSTC*; (d) Balanced-MSTC*.

k partitions of Π can be denoted by k key nodes

$$\mathcal{P} = \{\pi^*_{\rho_1}, \pi^*_{\rho_2}, \dots, \pi^*_{\rho_k}\} \quad (4)$$
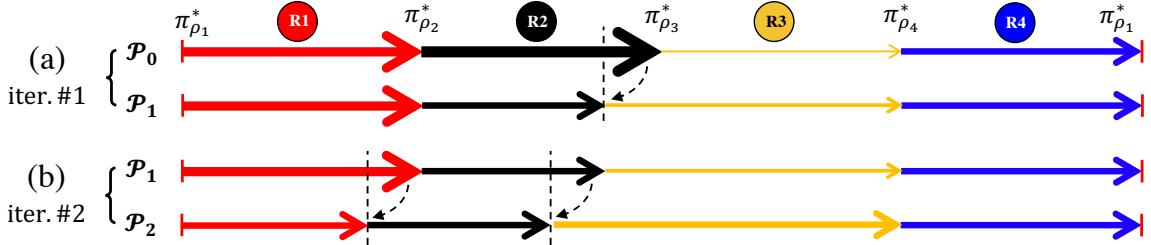
Fig. 6. Balanced-Cut Partition. (a) Given an initial partitions $\mathcal{P}_0$ generated by Naïve-MSTC*, we obtain $\Pi_{min} = \Pi_3$ and $\Pi_{max} = \Pi_2$. (b) Starting with $\mathcal{P}_1$ resulting from the first iteration, we have $\Pi_{min} = \Pi_3$ and $\Pi_{max} = \Pi_1$, and their in-between partition Π_2.

Here, we use π^* to represent a key node that separates two adjacent partitions. ρ_i is the node index from which a partition of coverage path starts. Then a partition is the coverage path Π_i consisting of all nodes between two key nodes.

$$\Pi_i = \{\pi^*_{\rho_i}, \pi_{\rho_i+1}, \pi_{\rho_i+2}, \dots, \pi^*_{\rho_{i+1}-1}\} \quad (5)$$

The main challenge of partitioning Π is to choose the key nodes. Non-backtracking MSTC use the depot to be the key nodes [28]. Each robot moves along its coverage path Π_i until it reaches its neighbor robot's depot, as shown in Fig. 5-(a). This partitioning strategy is inefficient due to the unbalanced workload. Even though backtracking optimization is suggested in [28], the resulting coverage path is still uneven. As shown in Fig. 5-(b), most of coverage paths are executed by robots $\mathcal{R}_1$ and $\mathcal{R}_4$.

B. Our MSTC Algorithm*

To solve the problem of unbalanced workload exhibiting in the partition strategy of MSTC, we proposed an improved MSTC, namely MSTC*.

Naïve-MSTC*: A straightforward solution is to generate k partitions, yielding the same amount of covering nodes in all the coverage paths Π_i (see Fig. 5-(c)). However, due to the lack of the coverage path costs, the resulting tasks for k robots may be still unbalanced.

Balanced-MSTC*: Here, we propose a balanced MSTC* algorithm, aiming at finding a set of well-balanced partitions $\mathcal{P}$ for Π with the consideration of coverage path costs. Given an initial set of workload partitions, our goal is to generate balanced workload for all robots by iteratively minimizing the maximum weights (refer to Eq. (1)). We elaborate our algorithm as follows.

Given an initial partition $\mathcal{P}_0$ generated using Naïve-MSTC*, the coverage path with the maximum weight is denoted by Π_{max} and it is partitioned by key node $\pi^*_{\rho_{max}}$. Analogically, the coverage path with the minimum weight is denoted by Π_{min} and it is partitioned by key node $\pi^*_{\rho_{min}}$. An example is shown in Fig. 6. For simplicity, we unfold a loop (i.e., the loop in Fig. 5) into line segments. An initial partition is $\mathcal{P}_0 = \{\pi^*_{\rho_1}, \pi^*_{\rho_2}, \pi^*_{\rho_3}, \pi^*_{\rho_4}\}$ for four robots $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$ and $\mathcal{R}_4$. Even though the nodes are evenly distributed, their weights are unbalanced. Here, a higher weight corresponds to a thicker line segment. As shown in Fig. 6-(a), the coverage path of $\mathcal{R}_2$ has the maximum

weight. That is, $\Pi_{max} = \Pi_2$. The coverage path of $\mathcal{R}_3$ has the minimum weight. Therefore, $\Pi_{min} = \Pi_3$. If Π_{max} and Π_{min} are adjacent, we iteratively remove the nodes at their boundary from Π_{max} and append them into Π_{min} until they are balanced. However, if Π_{max} and Π_{min} are not adjacent. We gradually shift the nodes from Π_{max} to Π_{min} through in-between partitions. An example is given in Fig. 6-(b), in which $\Pi_{max} = \Pi_1$ and $\Pi_{min} = \Pi_3$. In Fig. 6-(b), the in-between partition is Π_2. In other words, we remove the rightmost nodes from Π_1 and append them to the in-between partition Π_2. Meanwhile, we remove the rightmost nodes from Π_2 and append them to the partition Π_3. During the shifting, the workload of the in-between partitions remain unchanged.

To determine the potential shift partition nodes, a straightforward strategy is to linearly search Π_{min} and Π_{max}. However, linear search can be very computationally expensive. Therefore, we propose the strategy of *balanced cut* to accelerate the search. This strategy relies on binary search by iteratively updating search bound. In fact, finding the best partitions is a NP-hard problem. However, our strategy is a greedy algorithm to gradually approximate the optimal partitions. Our *balanced cut* algorithm is summarized as the pseudo-code in Algorithm 1. The Balanced-MSTC* algorithm is summarized as the pseudo-code in Algorithm 2.

C. MSTC under Limited Workload Capacity*

In this section, we explain the process of extending Balanced-MSTC* algorithm to handle limited workload capacity. Assume that robots have equal workload capacity c. Due to limited workload capacity, a robot needs to return to its depot for refilling. Let the traversal time spending on going back to depots and refilling be $r_i = \lceil \frac{\|\Pi_i\|}{c} \rceil$ for robot $\mathcal{R}_i$. Therefore, the total time spending on returning and refilling is $\sum_{i=1}^{k} r_i$.

If the capacity $c \geq \frac{\|\Pi\|}{k}$, $\forall i \in [1, k]$, we have $r_i = 1$. We can obtain k partitions without refilling. If the capacity $c < \frac{\|\Pi\|}{k}$, every robot needs to refill in order to accomplish the tasks. This will have additional $n = \sum_{i=1}^{k} r_i - k$ partition nodes in Π, which requires to be evenly distributed to k robots. More specifically, we can perform Balanced-MSTC* by assuming n robots. After obtaining n partitions, we merge some adjacent partitions to generate k new partitions for k

Algorithm 1: Balanced-Cut $(\mathcal{P}, \Pi_{min}, \Pi_{max})$

Input: A set of partitions $\mathcal{P}$, and Π_{min} and Π_{max}
Output: A set of balanced partitions

1 left $\leftarrow 0$
2 right $\leftarrow \|\Pi_{min}\| + \|\Pi_{max}\|$
3 $m = \lfloor \frac{\text{left+right}}{2} \rfloor$
4 **while** left $<$ right **do**
5     $s \leftarrow \left(\lfloor \frac{\text{left+right}}{2} \rfloor - m \right)$
6     $m \leftarrow \lfloor \frac{\text{left+right}}{2} \rfloor$
7   **for** $\Pi_i : \Pi_{min}, \Pi_{min+1}, , ..., \Pi_{max}$ **do**
8   shift partition node $\pi^*_{\rho_i}$ of Π_i by s nodes
9   update the coverage cost $\mathcal{W}_{\Pi_i}$ for Π_i
10   **end**
11   update $\mathcal{P}$
12   **if** $\mathcal{W}_{\Pi_{min}} < \mathcal{W}_{\Pi_{max}}$ **then**
13   left $\leftarrow (m+1)$
14   **else**
15   right $\leftarrow (m-1)$
16   **end**
17 **end**
18 **return** $\mathcal{P}$

Algorithm 2: Balanced-MSTC*$(\mathcal{G}, \mathcal{H}, k)$

Input: Covering graph $\mathcal{G}$, spanning graph $\mathcal{H}$ and k robots
Output: k coverage paths $\{\Pi_1, \Pi_2, ..., \Pi_k\}$

1 $\Pi \leftarrow$ Spiral-STC path for a single robot using $\mathcal{G}, \mathcal{H}$
2 $\mathcal{P} \leftarrow \{\pi^*_{\rho_i}\}$
3 determine Π_{max} and Π_{min}
4 compute their weights $\mathcal{W}_{\Pi_{max}}$ and $\mathcal{W}_{\Pi_{min}}$
5 **while** $\mathcal{W}_{\Pi_{max}} > \mathcal{W}_{\Pi_{min}}$ **do**
6   **for** $\Pi_i : \Pi_1, \Pi_2, , ..., \Pi_k$ **do**
7   compute weights $\mathcal{W}_{\Pi_i}$
8        $\mathcal{W}_{\Pi_{max}} = \max(\mathcal{W}_{\Pi_{max}}, \mathcal{W}_{\Pi_i})$
9        $\mathcal{W}_{\Pi_{min}} = \min(\mathcal{W}_{\Pi_{min}}, \mathcal{W}_{\Pi_i})$
10   **end**
11     $\Pi_{max} \leftarrow$ coverage path associating with $\mathcal{W}_{\Pi_{max}}$
12     $\Pi_{min} \leftarrow$ coverage path associating with $\mathcal{W}_{\Pi_{min}}$
13     $\mathcal{P} \leftarrow$ Balanced-Cut $(\mathcal{P}, \Pi_{min}, \Pi_{max})$
14 **end**
15 **return** $\Pi = \{\Pi_1, \Pi_2, ..., \Pi_k\}$ partitioned by $\mathcal{P}$ (Eq. 5)

---

robots

$$\Pi_i = \bigcup_{j=i \cdot \frac{n}{k}}^{(i+1) \cdot \frac{n}{k} - 1} \Pi'_j. \tag{6}$$

VI. RESULTS & ANALYSIS

We will now explain some implementation details of our algorithms and show our experimental results. Moreover, we compare our algorithms (Naïve-MSTC* and Balanced-MSTC*) against existing spiral-STC based methods: classic MSTC (MSTC-NB) [28], MSTC with backtracking (MSTC-BO) [28] and Multi-robot Forest Coverage (MFC) [29].

A. Implementation

We used DEM data from SRTM Digital Elevation Database of CGIAR[37] for steepness anaylysis and satellite map data from Sentinel-2 imagery of ESA[38] for DFC-classification. Then we incorporate them into the original terrain map to generate two traversability maps. In Eq. 2, we set $\alpha = \frac{1}{3}$ and $\beta = \frac{2}{3}$.

B. Regular Grid Map

We first tested our algorithm on small regular grid maps (see Figs. 7-(a) and (b)). The first regular grid map, shown in Fig. 7(a), is a unweighted blocked terrain used in [29], [33], [34]. Four robot depots are represented by stars and located at the lower-left cells. All four robots start from their
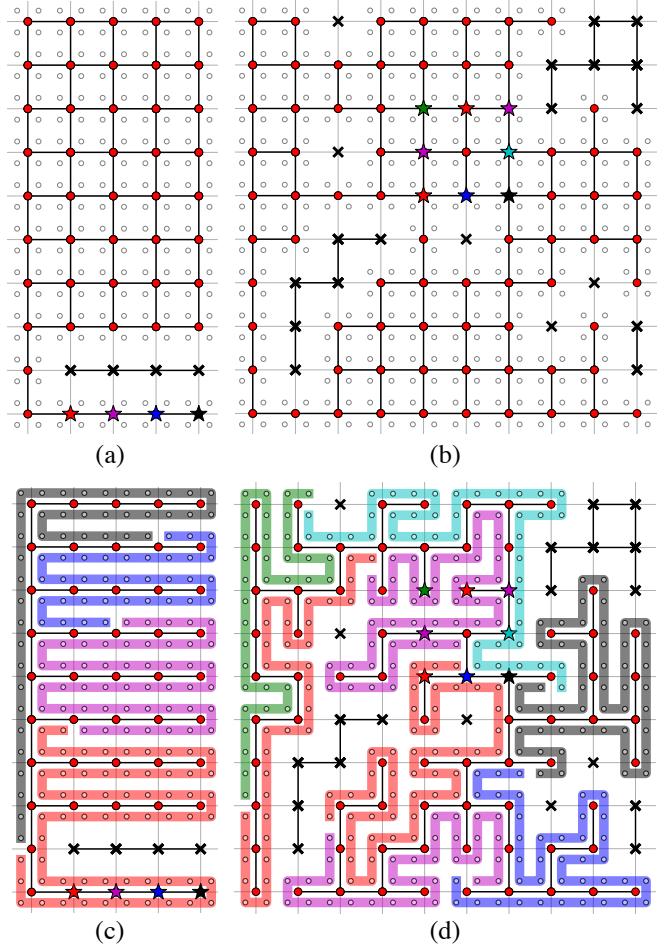

Fig. 7. Regular grid terrains. Top: terrains to be covered; Bottom: mCPP using our Balanced-MSTC*. (a)(c) k=4 and c=∞. (b)(d) k=8 and c=∞.

individual depot. The second scenario, shown in Fig. 7-(b), is a randomly generated 10×10 weighted terrain, with 8 robot depots and some random blocked cells. The blocked cells are indicated by $\times$ in these figures. The results of coverage paths are given in Figs. 7-(c) and (d).

Comparison: Fig. 8 shows the performance scalability of a few spiral-STC based methods with respect to the number of robots k and workload capacity c. We evaluated the performance improvement using the reduction ratio of

the maximum weights (refer to Eq. (1)). We also compared our MSTC* against MSTC-BO and MFC. our Balanced-MSTC* algorithm outperforms others. Especially, for the first terrain, Balanced-MSTC* has significant improvement against Naïve-MSTC*.
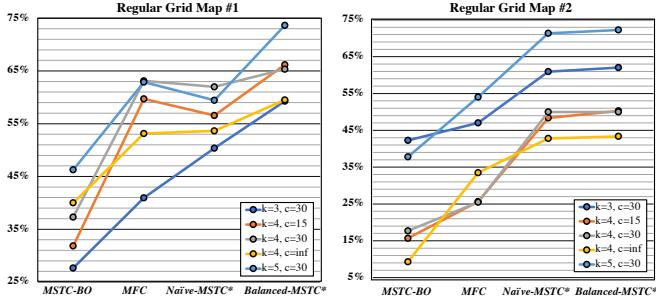

Fig. 8. Performance and scalability comparison for regular grid terrains in terms of the reduction ratio of the maximum weights.

C. Field Terrain

We also apply our algorithm to field terrains in the real-world applications, as shown in Figs. 9-(a) and (b). The
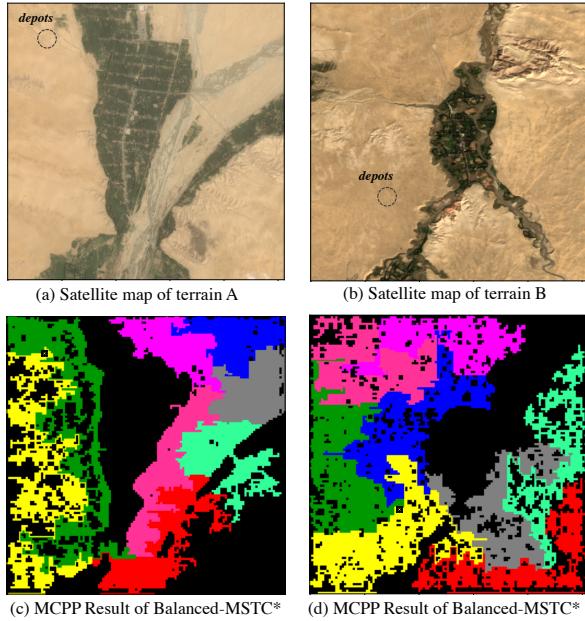

(a) Satellite map of terrain A   (b) Satellite map of terrain B

(c) MCPP Result of Balanced-MSTC*   (d) MCPP Result of Balanced-MSTC*

Fig. 9. Real Terrains. The robot depots are circled in the given map. (a) Satellite map at (77.88°E, 37.35°N); (b) Satellite map at (78.06°E, 37.26°N); (c)(d) The mCPP results using our Balanced-MSTC* algorithm ($k = 8$, $c = 400$).

mCPP results using our Balanced-MSTC* algorithm are given in Figs. 9-(c) and (d), respectively. The given satellite maps in Figs. 9-(a) and (b) are divided into 256×256 cells. After traversability processing (i.e., DFC-classification and steepness analysis), their spanning graphs include 10238 nodes and 17931 edges in Fig. 9-(a), and 12476 nodes and 19235 edges in Fig. 9-(b).

Comparison: We compare our algorithms against the spiral-STC based methods in terms of the reduction ratio

of the maximum weights. As shown in Fig. 10, our Naïve-MSTC* and Balanced-MSTC* outperform MSTC-BO and MFC.
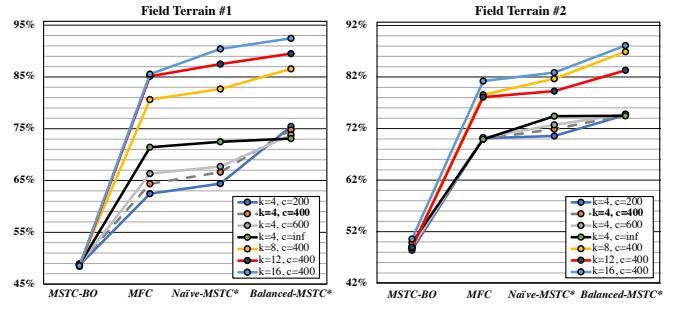

Fig. 10. Performance and scalability comparison for field terrains in terms of the reduction ratio of the maximum weights.

In addition, we further compare the scalability with respect to the number of robots k and workload capacity c. As shown in Fig. 10, MSTC-BO shows the worst scalability. In general, our Naïve-MSTC* and Balanced-MSTC* exhibit a higher scalability as the number of robots and the workload capacity increase. For example, the reduction ratio increases as the number of robots increases ($k = 4, 8, 12, 16$ for $c = 400$). However, we observed that the scalability of our algorithm is output sensitive. The performance improvement becomes less significant as the workload capacity continuously increases. For example, the performance of Balanced-MSTC* becomes similar to Naïve-MSTC* for $k = 4, c = \infty$ (see Fig. 10). Intuitively, our Balanced-MSTC* greatly benefits from the shortest path traversal during workload refilling. If refilling is unnecessary ($c=\infty$), the performance gain can be neglected.

VII. CONCLUSIONS & FUTURE WORK

We have presented an efficient algorithm MSTC* for multi-robot coverage path planning. Our algorithm improved spiral spanning tree coverage method by incorporating strict physical constraints like terrain traversability and material load capacity. We have performed extensive comparison with other mCPP methods both in regular grid maps and real-world terrains. Our method showed significant performance improvement against existing spiral-STC mCPP methods.

There are a few limitations in our algorithm. Our algorithm to find the partitions is greedy such that there is no guarantee to find the best. Our algorithm requires a few problem-dependent parameters such as α and β to compute the edge costs in covering graph $\mathcal{G}$ and spanning graph $\mathcal{H}$.

For future work, we would like to apply our techniques to real robots. Since the runtime communication and synchronization can cause overhead, we would like to investigate the extensions of our algorithm to de-centered environments.

REFERENCES

[1] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[2] F. Yasutomi, M. Yamada, and K. Tsukamoto, "Cleaning robot control," in *IEEE International Conference on Robotics and Automation*, 1988, pp. 1839–1841.

[3] S. Hert, S. Tiwari, and V. Lumelsky, "A terrain-covering algorithm for an AUV," in *Underwater Robots*, 1996, pp. 17–45.

[4] A. Ahmadzadeh, J. Keller, G. Pappas, A. Jadbabaie, and V. Kumar, "An optimization-based approach to time-critical cooperative surveillance and coverage with UAVs," in *Experimental Robotics*, 2008, pp. 491–500.

[5] E. U. Acar, H. Choset, Y. Zhang, and M. Schervish, "Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods," *International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 441–466, 2003.

[6] M. Ollis and A. Stentz, "Vision-based perception for an automated harvester," in *IEEE/RSJ International Conference on Intelligent Robot and Systems*, 1997, pp. 1838–1844.

[7] L. Heng, A. Gotovos, A. Krause, and M. Pollefeys, "Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 1071–1078.

[8] L. Lin and M. A. Goodrich, "UAV intelligent path planning for wilderness search and rescue," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 709–714.

[9] M. Bosse, N. Nourani-Vatani, and J. Roberts, "Coverage algorithms for an under-actuated car-like vehicle in an uncertain environment," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 698–703.

[10] X. Zhang. Tree planting robots for ecological restoration and desert afforestation. Youtube. [Online]. Available: https://www.youtube.com/watch?v=oGaYE4gTrvE

[11] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and Service Robotics*, London, 1998, pp. 203–209.

[12] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*, 2007.

[13] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *The International Journal of Robotics Research*, vol. 21, no. 4, pp. 331–344, 2002.

[14] H. Choset, "Coverage for robotics–a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1-4, pp. 113–126, 2001.

[15] Y. Gabriely and E. Rimon, "Spiral-STC: an on-line coverage algorithm of grid environments by a mobile robot," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2002, pp. 954–960.

[16] C. Luo, S. X. Yang, D. A. Stacey, and J. C. Jofriet, "A solution to vicinity problem of obstacles in complete coverage path planning," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2002, pp. 612–617.

[17] S. X. Yang and C. Luo, "A neural network approach to complete coverage path planning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 1, pp. 718–724, 2004.

[18] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Optimal complete terrain coverage using an unmanned aerial vehicle," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 2513–2519.

[19] W. H. Huang, "Optimal line-sweep-based decompositions for coverage algorithms," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2001, pp. 27–32.

[20] P. A. Jimenez, B. Shirinzadeh, A. Nicholson, and G. Alici, "Optimal area covering using genetic algorithms," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2007, pp. 1–5.

[21] S. Tully, G. Kantor, and H. Choset, "Leap-frog path design for multi-robot cooperative localization," in *Field and Service Robotics*, 2010, pp. 307–317.

[22] T. Bretl and S. Hutchinson, "Robust coverage by a mobile robot of a planar workspace," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 4582–4587.

[23] R. Almadhoun, T. Taha, L. Seneviratne, and Y. Zweiri, "A survey on multi-robot coverage path planning for model reconstruction and mapping," *SN Applied Sciences*, vol. 1, 07 2019.

[24] I. Hameed, "Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain," *Journal of Intelligent and Robotic Systems*, vol. 74, 06 2013.

[25] I. Shnaps and E. Rimon, "Online coverage of planar environments by a battery powered autonomous mobile robot," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 425–436, 2016.

[26] M. Wei and V. Isler, "Coverage path planning under the energy constraint," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 368–373.

[27] C. Wu, C. Dai, X. Gong, Y. Liu, J. Wang, X. D. Gu, and C. C. L. Wang, "Energy-efficient coverage path planning for general terrain surfaces," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2584–2591, 2019.

[28] N. Hazon and G. A. Kaminka, "Redundancy, efficiency and robustness in multi-robot coverage," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 735–741.

[29] Xiaoming Zheng, Sonal Jain, S. Koenig, and D. Kempe, "Multi-robot forest coverage," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 3852–3857.

[30] N. Agmon, N. Hazon, and G. A. Kaminka, "Constructing spanning trees for efficient multi-robot coverage," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 1698–1703.

[31] N. Agmon, N. Hazon, and G. A. Kaminka, "The giving tree: Constructing trees for efficient offline and online multi-robot coverage," *Annals of Mathematics and Artificial Intelligence*, vol. 52, no. 2–4, p. 143–168, 2008.

[32] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha, "Min–max tree covers of graphs," *Operations Research Letters*, vol. 32, no. 4, pp. 309 – 315, 2004.

[33] X. Zheng and S. Koenig, "Robot coverage of terrain with non-uniform traversability," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 3757–3764.

[34] X. Zheng, S. Koenig, D. Kempe, and S. Jain, "Multirobot forest coverage for weighted and unweighted terrain," *IEEE Transactions on Robotics*, vol. 26, no. 6, pp. 1018–1031, 2010.

[35] A. Sipahioglu, G. Kirlik, O. Parlaktuna, and A. Yazici, "Energy constrained multi-robot sensor-based coverage path planning using capacitated arc routing approach," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 529 – 538, 2010.

[36] X. Huang, M. Sun, H. Zhou, and S. Liu, "A multi-robot coverage path planning algorithm for the environment with multiple land cover types," *IEEE Access*, pp. 1–1, 2020.

[37] R. H. amd A. Nelson and A. Jarvis, "Hole-filled seamless SRTM data v4, International Centre for Tropical Agriculture (CIAT)," http://srtm.csi.cgiar.org, 2008, accessed: 2020-09-30.

[38] M. Drusch, U. Del Bello, S. Carlier, O. Colin, V. Fernandez, F. Gascon, B. Hoersch, C. Isola, P. Laberinti, P. Martimort, *et al.*, "Sentinel-2: ESA's optical high-resolution mission for GMES operational services," *Remote sensing of Environment*, vol. 120, pp. 25–36, 2012.

[39] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *European Conference on Computer Vision*, 2018, pp. 801–818.

[40] M. Schmitt, J. Prexl, P. Ebel, L. Liebel, and X. X. Zhu, "Weakly supervised semantic segmentation of satellite images for land cover mapping – challenges and opportunities," *arXiv (to appear)*, 2020.

[41] M. Schmitt, L. H. Hughes, C. Qiu, and X. X. Zhu, "SEN12MS - A curated dataset of georeferenced multi-spectral Sentinel-1/2 imagery for deep learning and data fusion," *CoRR*, 2019. [Online]. Available: http://arxiv.org/abs/1906.07789

[42] M. S. L. H. P. G. N. Y. R. Hansch, "IEEE GRSS data fusion contest," 2019. [Online]. Available: http://dx.doi.org/10.21227/rha7-m332