MDCPP: Multi-robot Dynamic Coverage Path Planning for Workload Adaptation

Jun Chen and Shinkyu Park

King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia,
jun.chen.1, shinkyu.park@kaust.edu.sa

Abstract. Multi-robot coverage path planning (MCPP) addresses the problem of computing paths for multiple robots to effectively cover entire area of interest. Conventional approaches for MCPP find such paths assuming that the robots need to move at fixed velocities. However, such assumption may not be viable in many real-world applications where each robot needs to adapt its velocity depending on an assigned coverage task. Hence, conventional approaches would unequally distribute coverage workloads across multiple robots and require longer time to cover the entire area. In this paper, we propose a novel multi-robot dynamic coverage path planning (MDCPP) algorithm for complete coverage of a two-dimensional space that dynamically assigns coverage areas to a team of robots. Especially, after an initial assignment of coverage areas based on robots' coverage capacities, the algorithm repeatedly swaps assigned areas to balance coverage workloads across the robots and plans optimal coverage paths within assigned areas. We also investigate a distributed algorithm to implement MDCPP over a range-constrained robotic network. Through simulations, we validate the efficacy of MDCPP qualitatively, demonstrate that it outperforms an existing sweeping algorithm, and quantify the effect of inter-robot communication range on the coverage performance.

Keywords: Coverage path planning, distributed sensor network, multi-robot coordination

1 Introduction

Coverage path planning (CPP) is the problem of computing paths that allow mobile robots to efficiently cover an area of interest. CPP has attracted considerable attention over the last decade due to its practical importance in applications such as target search and inspection [9].

CPP algorithms use simple yet effective coverage patterns, e.g., back-and-forth and spiral sweeping, while optimizing various design objectives such as travel distance and time, energy consumption, etc [14]. Back-and-forth sweeping approaches decompose the coverage region to determine a sequence of subregions for a single robot to cover, and generate a zigzag path traversing each subregion [13, 20]. Spiral sweeping approaches instead generate a circular path for a robot to cover each subregion from the outside toward the inside [3, 8]. Multi-robot CPP (MCPP) extends CPP often by adopting area decomposition methods to divide the coverage area and assign a task of covering each

subregion to a robot. Cellular decompositions [7, 11, 12, 15] partition the area into non-overlapping cells to allocate each cell to a robot, while grid-based decompositions [10, 16, 17, 19] divide the area into grid cells and assign each cell to a robot which then creates a spanning tree to circumnavigate around its assigned cell. Recently, Collins et al. [5] propose a time-efficient MCPP solution that assigns balanced workloads to a team of robots using grid-based weighted Voronoi decomposition and the robots' initial positions. The approach of Tolstaya et al. [18] trains a graph neural network controller for a robot team to cover a waypoint graph which connects all vertices to be explored.

Most of the existing MCPP approaches assume that the robots are required to move at constant velocities at all time for which a single and static area partitioning is sufficient for effective workload balancing across multiple robots. However, in many real-world scenarios, each robot needs to adapt its moving speed depending on difficulties in covering assigned areas. Hence, this paper aims at developing a novel MCPP algorithm that dynamically distributes workloads of multiple robots while they are carrying out assigned coverage tasks. The new approach would leverage heterogeneity in multi-robot systems for area coverage and allow the workload distribution to be adaptive to changes in coverage and target detection progress, robots' power level, and system-level changes such as loss of inter-robot communication and failure of individual robots.

We propose a novel algorithm, namely the multi-robot dynamic coverage path planning (MDCPP) algorithm, and its distributed implementation. For the workload distribution, MDCPP relies on capacity-constrained Voronoi diagrams [2] over a finite space to partition the coverage area and adopts a grid-based representation of it. Then the shortest path to iterate through each grid once in each partition is generated for complete coverage. Coverage area partitioning is optimized at each time step based on robots' current coverage capability and workload capacity, and a sub-optimal goal is selected for each robot in its assigned workspace. Through simulations, we demonstrate that the new method achieves significant improvements in total coverage time especially when the time required for each robot to cover its assigned area is highly unpredictable.

2 Problem Formulation

We begin with illustrating a motivating example of coral reef monitoring. A team of unmanned underwater vehicles (UUVs) equipped with cameras are tasked with exploring a certain area on sea bed, collect images of various coral species, and restore bleached corals at appropriate locations based on analysis of collected images. While UUVs can move at the maximum speed to desired locations, they must move at lower speed to collect high-resolution images, and move at the lowest speed or even stop to restore bleached corals. The maximum speed may vary due to ocean currents and battery level, the image-collection speed may vary based on light condition, and the speed of coral restoration may vary due to the shape and hardness of the ocean surface. Our goal is to design an algorithm that dynamically assigns monitoring/manipulation tasks to the UUV team to minimize the time it takes to complete the tasks despite the unpredictability of the tasks.

To formalize, we denote a team of num_r robots as $R = \{r_1, \ldots, r_{num\_r}\}$. Each robot is equipped with an isotropic sensor with sensing range s_range and are able to localize

themselves. Note that our method also applies to anisotropic sensors by mapping the detection models to equivalent isotropic ones [4]. Robots' maximum communication range is denoted by c_range. Each robot r_i moves at three different velocities on need basis: the maximum velocity $v_{max,i}$, the detecting velocity $v_{det,i}$, and the interacting velocity $v_{int,i}$. All three velocities $v_{max,i}$, $v_{det,i}$, and $v_{int,i}$ are random variables, with means denoted by $\bar{v}_{max,i}$, $\bar{v}_{det,i}$, and $\bar{v}_{int,i}$, respectively.

Consider a known convex task space $E_0 \subset \mathbb{R}^2$ in which a number of stationary targets of interest are positioned within a set $T \subset E_0$ of unknown locations.

Definition 1 (Coverage). *An area $A \subset E_0$ is said to be covered if any point $p \in A$ is* detected *by at least one robot for possible targets , and necessary* interactions *are performed by at least one robot for each target within A.*

At each time step t, we denote as E_t uncovered areas of E_0, where E_t is partitioned into num_g uniform square grids $G = \{g_1, \ldots, g_{num\_g,t}\}$ such that $g_1 \cup \cdots \cup g_{num\_g,t} = E_t$. We assume that each g_i is the largest square that a robot can cover. With abuse of notation, we use $g_1, g_2, \ldots$ to denote the centroid of corresponding squares. To define each robot's coverage task, we divide E_t into num_r subregions $S_{1,t}, \ldots, S_{num\_r,t} \subset E_0$, where $S_{i,t}$ is the area that each robot r_i is tasked to cover. We define the complete coverage task as follows.

Definition 2 (Complete Coverage). *The robot team R* covers *all grids in E_0.*

Remark 1. In order for a grid to be *covered*, it needs to be detected for targets (with speed $v_{det,i}$) once by one robot $r_i \in R$ since targets are stationary, and each target found in the grid is needed to be interacted with (with speed $v_{int,i}$) once by one robot $r_i \in R$. Once a grid is *covered*, any robot $r_j \in R$ can move with speed $v_{max,j}$ if it passes through the grid without slowing down to detect the grid for targets or to interact with targets.

Remark 2. A robot r_i cannot detect or interact with targets located inside a grid when passing through it (with speed $v_{max,i}$). Therefore, passing through a grid is not considered as covering the grid.

At time step t, the workload of robot r_i at its current location $q_{i,t} \in S_{i,t}$ is defined as the time it takes for robot r_i to cover its assigned $S_{i,t}$, given by $w_{i,t} = \mathcal{W}(S_{i,t}, q_{i,t})$. Since the number of targets and their locations are unknown to robots, $w_{i,t}$ is approximated by

$$w_{i,t} = \alpha_i \sum_{j=1}^{num\_g_i} \|q_{i,t} - g_j\|, \quad g_j \in S_{i,t}, \tag{1}$$

where num_g_i is the number of grids in $S_{i,t}$ such that $\sum_{i=1}^{num\_r} num\_g_i = num\_g$, α_i is a weight associated with r_i's coverage capability, and $\|\cdot\|$ is the Euclidean norm. Here, α_i is a tuning parameter that reflects the estimated coverage capabilities considering traveling, detecting and interacting speeds of robots in a heterogeneous team, though the capabilities sometimes cannot be precisely quantified. For example, we may consider two robots r_i and r_j to have identical αs if $v_{max,i}$ is slightly greater than $v_{max,j}$ and $v_{det,i}$ is slightly less than $v_{det,j}$. Homogeneous robots have identical αs.

Remark 3. This paper mainly focuses on the dynamic workload adaptation scheme. Therefore, for simplicity, E_0 is assumed convex for simplicity, and the cost of a robot moving to a goal is quantified by the Euclidean norm. However, our algorithms can be applied to obstructed environments by replacing the Euclidean norm to the actual length of a feasible path for a robot to reach a goal.

Thereby, maximizing coverage efficiency of the robot team can be achieved by solving the following optimization.

Problem 1 (Optimized Grid Assignment).

$$\text{minimize} \quad \max_{i\in\{1,\cdots,num\_r\}} w_{i,t}. \tag{2}$$

For simplicity, the rest of the paper neglects subscript t unless otherwise specified.

3 Dynamic Coverage Path Planning

To solve Problem 1, we adopt the following three steps. Firstly, the robots compute their initial locations to move offline using Lloyd's algorithm. Secondly, the robots dynamically partition E_t by assigning each grid $g \in E_t$ to the closest robot satisfying a workload capacity constraint. Thirdly, each robot determines its next location to cover within its assigned area. In what follows, we explain algorithms detailing each of the above steps.

3.1 Determine Initial Goals

To maximize the coverage efficiency, the robots should initially spread out across the coverage area. To achieve this, the robots determine their initial locations using Lloyd's algorithm. We first consider the case where robots have identical coverage capabilities. Let $\mathcal{S} = \{S_1, \ldots, S_{num\_r}\}$ and $Q = \{q_1, \ldots, q_{num\_r}\}$. The objective of Lloyd's algorithm is to minimize the following functional:

$$\mathcal{H}(Q, \mathcal{S}) = \sum_{i=1}^{num\_r} \sum_{j=1}^{num\_g_i} f\big(\|g_j - q_i\|\big)\phi(g), \tag{3}$$

with respect to both partition set $\mathcal{S}$ and robot positions Q, where $g \in E_0$ is a grid in E_0 and $g_j \in S_i$ is an assigned grid of r_i, $f(\cdot)$ is be a monotonically increasing function, and $\phi(g)$ is a density function over E_0. We select to use a uniform distributed $\phi(g)$ over E_0. The quantity $f\big(\|g_j - q_i\|\big)$ represents the cost of traveling the distance of $\|g_j - q_i\|$. Minimizing $\mathcal{H}$ with respect to $\mathcal{S}$ induces a partition of the environment $V_i = \{g_j \mid i = \arg\min_{k=1,\ldots,num\_r} \|g_j - q_k\|\}$. In other words, V_i is the collection of all grids that are the nearest neighbor of r_i. This is the Voronoi partition, and these V_i are the Voronoi cells, which are convex by construction. Minimizing $\mathcal{H}$ with respect to Q leads each robot to the weighted centroid of its Voronoi cell [6], that is

$$q_i^* = \frac{\sum_{j=1}^{num\_g_j} g_j\, \phi(g_j)}{\sum_{j=1}^{num\_g_j} \phi(g_j)}. \tag{4}$$

Algorithm 1: Goal Initialization

```
1 while true do
2     Construct V using Q
3     for Each robot r_i do
4         Find the centroid q_i^* of V_i using Equation (4)
5         Δq_i ← ‖q_i^* − q_i‖
6         Move to q_i^*
7     if ‖Δq_i − q_i^*‖ < ϵ for all r_i then
8         Break
```

Algorithm 1 outlines the goal initialization step using Lloyd's algorithm, where ϵ in line 7 is a constant determining the stopping criterion. By controlling each robot r_i towards q_i^*, the team is spread out towards initial positions where the total cost (3) of traveling to all points in E is minimized.

When robots' coverage capabilities αs are heterogeneous, Lloyd's algorithm is implemented with power diagrams. The power diagram is a variant of the standard Voronoi diagram that uses the power distance, $f(\|g - q_i\|) = \|g - q_i\|^2 - \rho_i^2$, where $\rho_i := \alpha_i$ is the power radius of r_i, instead of using Euclidean distance in Equation (3). As a result, the robots with better coverage capabilities move further away from others in this initialization step.

3.2 Grid Assignment

The goal of the grid assignment algorithm is to dynamically assign all un-searched grids to the robot with the lowest cost to reach it under capacity constraints $w_{1,t} = \cdots = w_{num\_r,t}$ for optimized coverage area assignment. This is similar to the goal of Algorithm 1 in [2] that generates capacity-constrained Voronoi tessellations. Inspired by Algorithm 1 in [2], we develop a distributed grid assignment algorithm, outlined in Algorithm 2.

Each robot r_i must have a unique ID number i. At the beginning, r_i must first search for its neighbor set $\mathcal{N}_i$, i.e., the set of all robots that are within its communication range. Then robot r_i compares its ID i with all other robots' IDs in $\mathcal{N}_i$. If robot r_i is with the smallest ID in its neighborhood, it requests $S_{j,t}$ from all neighbors $r_j \in \mathcal{N}_i$, initializes a random assignment for each neighbor and itself that fulfills the coverage capability constraints αs, and sends the initial assignment to each neighbor, outlined by lines 4-8. In another word, for each robot, the number of assigned grids is proportional to its coverage capacity α. After that, robot r_i requests current locations and grid assignments from neighbors with greater ID values, computes and send back the swapped assignments to them, as outlined by lines 9-23 in Algorithm 2. To swap assignments, each robot in a pair inserts its assigned grids into a heap data structure, along with a key that quantifies the distance that the paring robot is closer to this grid, given by

$$\Delta e(g, q_i, q_j) = \|g - q_i\| - \|g - q_j\|. \tag{5}$$

where $\|\cdot\|$ is the Euclidean norm. Here, a positive $\Delta e(g, q_i, q_j)$ value indicates that robot r_i can optimize its grid assignment by swapping g with robot r_j, while a negative one reveals that g is assigned to the closest robot between r_i and r_j. As noted in Remark 3, the Euclidean norms can be replaced by actual length of a planned path for an occluded environment. Afterwards, each pair of robots swap all grids that leads to a positive Δe until all grids are assigned to the closest robot in each pair and optimal assignments are returned. Meanwhile, r_i send its current location and grid assignment to neighbors with smaller ID values and request computed swapped assignments from them.

Algorithm 2 yields identical assignments to the results of Algorithm 1 in [2] if communication range is unlimited, or if all robots are connected through a communication graph with each other directly or indirectly. Therefore, Algorithm 2 may or may not produce a globally optimal grid assignment depending on the communication constrain, but yields optimal assignments locally. We also remark that the computational burdens are not equivalent among teams as robots with smaller ID values compute more swapped assignments compared to these with greater ID values. Thus, in practice, heterogeneous robots can be labeled based on its computational capability, energy capacity, etc.

3.3 Coverage Path Planning

Once a robot is assigned the best grids to cover at each time step, it must plan the trajectory to cover the assigned grids.To execute the area coverage, the robot r_i is to find a shortest path $P_i = g_1, \ldots, g_{num\_g_i}$ to traverse all num_g grids $g \in S_i$, which is formulated as follows.

Problem 2 (Coverage Path Planning).

$$\min \sum_{k=1}^{num\_g_i} \sum_{l \neq k, l=1}^{num\_g_i} \text{dist}(g_k, g_l) \cdot x_{kl}, \tag{6}$$

$$\text{s.t.} \sum_{k=1}^{l-1} x_{kl} + \sum_{k=l+1}^{num\_g_i} x_{lk} = 2, \text{for all} j; \tag{7}$$

$$\sum_{k,l \in Sub}^{num\_g_i} x_{kl} \leq |Sub| - 1, \text{for each subset } Sub. \tag{8}$$

In Equation (6), dist$(\cdot)$ is the cost of move from one grid to another, which is associated with distances between two grids. We use Euclidean distance since E_0 is assumed a convex open area, while other measures can be applied in non-convex environment such as the shortest distance to reach a point. The starting grid $P_i(1)$ of the path is set to be the center of the grid that r_i is located. Equation (7) ensures a grid can only be visited once and Equation (8) ensures that the planned path is connected, where Sub is the set of all subsets of S_i and $|\cdot|$ is the number of values in a set. Problem 3 is the symmetric case of the traveling salesman problem (TSP). We apply the nearest neighbor algorithm [1], a heuristic algorithm to solve it due to the computational efficiency.

In order to dynamically balance the workloads team-wise online, a robot r_i repartition the space with neighbors once the size of $S_{i,t}$ is less than a small n_0 and r_i is

Algorithm 2: Distributed grid assignment

for Each robot r_i R with ID i **do**
  $stable_i \leftarrow false$
  Find neighbors $\mathcal{N}_i$
  if i is the smallest ID in r_i's neighborhood **then**
    r_i requests $S_{j,t}$ from all neighbors $r_j \in \mathcal{N}_i$ and initializes $S_{i,t}$, $S_{j,t}$
    r_i sends $S_{j,t}$ to all neighbors $r_j \in \mathcal{N}_i$
  else
    r_i broadcasts $S_{i,t}$
  while $stable_i = false$ **do**
    $stable_i \leftarrow true$
    for All robots in $\mathcal{N}_i$ with ID j **do**
      if $i < j$ **then**
        Request $S_{j,t}, q_j$ from r_j
        Initialize two heap data structure H_i, H_j
        for Each grid $g_i \in S_{i,t}$ **do**
          Insert g_i into H_i with $\Delta e(g_i, q_i, q_j)$ as its key
        for Each grid $g_j \in S_{j,t}$ **do**
          Insert g_j into H_j with $\Delta e(g_j, q_j, q_i)$ as its key
        while $\|H_i\| > 0$ and $\|H_j\| > 0$ and $max(H_i) + max(H_j) > 0$ **do**
          Swap assignment of grids g_i and g_j
          Remove $max(H_i)$ from H_i and $max(H_j)$ from H_j
          $stable_i \leftarrow false$
        Send $S_{j,t+1}$ to r_j as r_j's assignment
      else
        Send S_j, q_j to r_j
        Request $S_{j,t+1}$ from r_j as r_j's assignment
return $\mathcal{S}_{t+1}$

Algorithm 3: MDCPP

```
1  Initialize goals for all robots using Algorithm 1 (offline)
2  Initialize grid assignment using Algorithm 2 (offline)
3  for Each robot r_i do
4      Move to the initial goal at v_{max,i}
5      Plan a coverage path P_i by solving Problem 3
6      while true do
7          if Re-partition requested by r_j ∈ N_i or |S_{i,t}| < n_0 then
8              Update S_{i,t} using Algorithm 2
9              Plan a coverage path P_i by solving Problem 3
10         else
11             if q_i is within grid P_i(k) then
12                 if Targets of interest are found in P_i(k) then
13                     v_i = v_{int,i}
14                 else
15                     v_i = v_{det,i}
16             else
17                 v_i = v_{max,i}
18             Move along P_i with speed v_i
```

about to complete covering its current assigned grids. Here, n_0 is a tuning parameter, and a greater n_0 results in more frequent data exchange and re-assignment of grids, consuming more computational resources, while being more resilient to possible intermittent communication lost. The overall MDCPP algorithm is outlined in Algorithm 3. Each robot moves to its initialized goal with the traveling velocity and plan a coverage path P_i in its initialized assigned grids, outlined in lines 4-5. Then each robot sequentially moves to each grid in P_i, detects targets in it, and interacts with them if targets are found, outlined in lines 11-18. Re-assignment of grids and re-planning of coverage path is required if r_i or any of its neighbors is about to complete its current path soon, outlined in lines 7-9.

4 Simulations

In this section, we demonstrate the efficacy of MDCPP through a series of simulations. In these simulations, environment E_0 is a squared area meshed with 20×20 squared grids. Each of the 400 grids is $10\,\text{m} \times 10\,\text{m}$. Unknown targets are distributed statically in some of these grids. Four robots are utilized to cover E_0, each equipped with an isotropic sensor with $s\_range = 7.07\,\text{m}$ so that the detection range is circumscribed on a grid. The average velocities of traveling, detection, and interaction are displayed on Table 1. Considering unexpected factors in practical applications, robots move at $50\% - 150\%$ of their average velocities for all three states. Four robots are considered to have the same coverage capability α since each of them has its own strengths in traveling, detection, or

Table 1. Average Velocities (m/s) of Each Robot

Velocities / Robot	traveling ($\bar{v}_{max}$)	Detection ($\bar{v}_{det}$)	Interaction ($\bar{v}_{int}$)
1	3.0	1.0	0.1
2	2.0	0.8	0.12
3	5.0	0.6	0.08
4	4.0	0.12	0.06

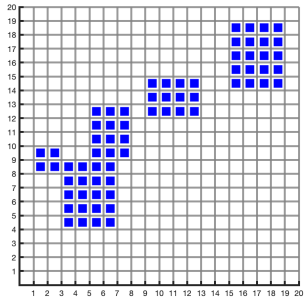


Fig. 1. Figure shows the simulation environment meshed with 20×20 grids. Static targets are located inside blue grids.

interaction and none of them is significantly superior to others, as revealed by Table 1. We set n_0 to 2. The MDCPP algorithm is run at 10 Hz.

4.1 Single Case

We first present a single case in which targets are distributed in blue areas in Figure 4. All four robots begin to move from the lower-left corner of E_0. Robots move to initial goals in 75 s and partition the space by assigning the grids, as shown in Figure 3a. Then robots generate their initial coverage paths and move along them to detect and interact with targets. Since no targets are distributed within lower-right quarter of E_0 (referred to Figure 4), the lower-right robot is about to complete traversing its coverage path ahead of others at 1294 s, as depict in Figure 3b. In order to balance the workloads, the team re-assigns the un-searched grids using Algorithm 2, shown as Figure 3c, and each robot re-plans its coverage path in the newly assigned area. At 2691 s, the upper-left robot is close to completing coverage and another re-assignment is conducted, shown as Figures 3d and 3e. Such process repeated until all grids in E_0 are covered at 3174 s, shown in Figure 3f. It can be seen that each robot roughly covers a clustered shape of area in E_0. The robot with a green trajectory covers the largest area mainly in the bottom-right quarter, and the robot with a light blue trajectory covers the smallest. The outcome is consistent with the target distribution shown in Figure 4 since targets are mostly distributed in the bottom-left quarter and the upper-right quarter, though robots' velocities also affect covered areas of robots. Although the workload for each robot is distinct and unpredictable, and the robot's velocities are uncertain, each robot dynamically balances the assigned

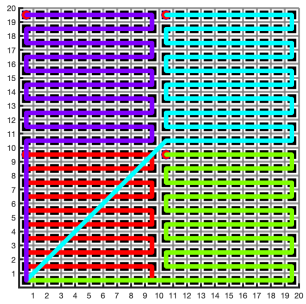


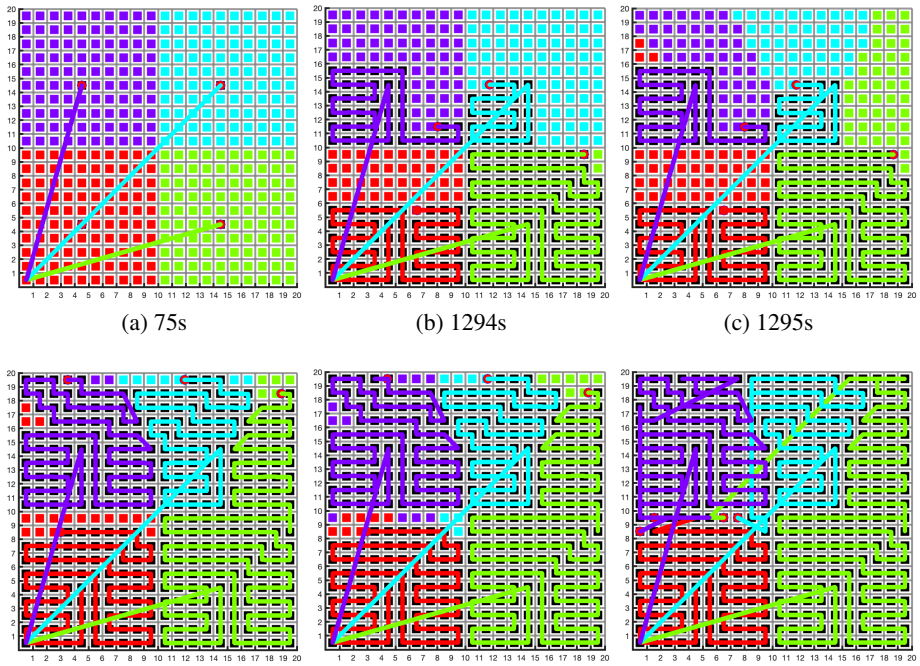
Fig. 2. Figure shows the trajectories of four robots after complete coverage of E_0 using the sweeping algorithm. Red, green, purple, and light blue curves display the trajectory of each robot, respectively.

workload online and covers its assigned space simultaneously to take advantage of all robots in the team.

4.2 Batch of Runs

We then run a batch of simulations for quantitative results. We compare MDCPP with the "sweeping" algorithm, a widely adopted coverage strategy that allows robots to divide E_0 into four identical rectangular areas based on robots' identical coverage capability. To the best of our knowledge, none of the existing work addresses the problem of MCPP with the unknown workload, so the sweeping algorithm is a standard comparison. In each assigned coverage area, each robot moves to the lower-left corner of the assigned area with traveling speed, zigzags with detecting speed to search each grid for targets, and switches to interacting speed once targets are found in a grid. By applying the sweeping algorithm, robots do not exchange information with each other, and there is no dynamic balance of workloads during coverage. The assignment of E_0 is time-invariant, and robots move in deterministic trajectories shown in Figure 4. We run 100 trials for the sweeping algorithm, and MDCPP algorithm with varying communication ranges c_range, ranging from 80 m to unlimited distance, respectively. For each test, the identical 100 target distributions are applied. For each of them, targets are distributed randomly in different locations while the total number of grids with target distributed are identical to Figure 4, meaning that the total workloads are identical among trials.

Figure 4 plots the time and summation of distances the team moves after completely covering E_0 in each of the six test scenarios. It is shown that the MDCPP algorithm reduces coverage time by over 30% compared to the sweeping algorithm, despite the fact that MDCPP leads robots to move longer total distances since idle robots may travel across regions to assist busy robots in order to dynamically balance workloads. We also find that as $\sum_1^{num\_r} \pi \cdot c\_range_i^2$ is getting closed to E_0, the communication limit shows no significant impact on the performance of MDCPP, revealing the robustness of our method under moderate communication constraints.



(a) 75s (b) 1294s (c) 1295s

(d) 2691s (e) 2692s (f) 3174s

Fig. 3. Figures show coverage procedures over a singer trial using the MDCPP algorithm. Robots are plotted by red disks. Grids are in four colors: red, green, purple, and light blue, depicting their assignments to the four robots. Black grids indicates the selected goals, which are excluded from E_t. Robots' traversed trajectories are plot by curves in same colors of their assigned grids.

5 Conclusions

This paper addresses a novel MCPP problem that considers undetermined and time-varying coverage velocities as opposed to conventional approaches which assume that the robot speeds are fixed. We propose the MDCPP algorithm that enables a distributed team of robots to cover a planet space quickly despite the velocities of robots being unknown and time-varying. The proposed strategy spreads the team across the coverage area using Lloyd's algorithm, dynamically partitions the coverage area based on robots' coverage capabilities, and plans an optimal coverage path for each robot in its assigned coverage area. As a result, workloads are dynamically balanced among individuals and the team is able to take advantage of all agents in order to complete the coverage task more quickly and efficiently. Simulation results reveal the effectiveness of MDCPP compared to the sweeping algorithm even when robots are under moderate communication distance constraints. Future work includes developing intermittent communication schemes for robots with extreme short communication ranges, and extending experiments to real-world scenarios.

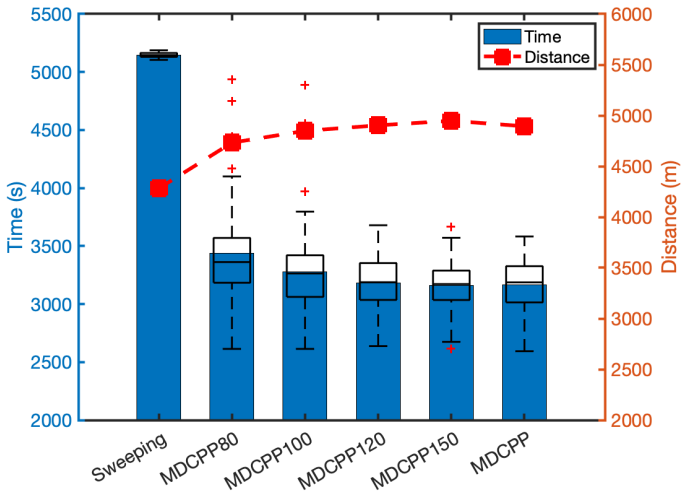


Fig. 4. Figure shows the bar plot of the average time and box plot for complete coverage over 100 trials in six test scenarios: sweeping algorithm ("Sweeping"), MDCPP algorithm with c_range equal to 80 m ("MDCPP80"), 100 m ("MDCPP100"), 120 m ("MDCPP120"), 150 m ("MDCPP150"), and MDCPP algorithm with no communication constraint ("MDCPP"), respectively. Average total distances the team move after completing coverage in each test scenario are also plot.

References

1. Applegate, D.L., Bixby, R.E., Chvátal, V., Cook, W.J.: The traveling salesman problem. In: The Traveling Salesman Problem. Princeton university press (2011)
2. Balzer, M., Schlömer, T., Deussen, O.: Capacity-constrained point distributions: A variant of lloyd's method. ACM Transactions on Graphics (TOG) **28**(3), 1–8 (2009)
3. Cabreira, T.M., Di Franco, C., Ferreira, P.R., Buttazzo, G.C.: Energy-aware spiral coverage path planning for uav photogrammetric applications. IEEE Robotics and automation letters **3**(4), 3662–3668 (2018)
4. Chen, J., Dames, P.: Distributed multi-target tracking for heterogeneous mobile sensing networks with limited field of views. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 9058–9064. IEEE (2021)
5. Collins, L., Ghassemi, P., Esfahani, E.T., Doermann, D., Dantu, K., Chowdhury, S.: Scalable coverage path planning of multi-robot teams for monitoring non-convex areas. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 7393–7399. IEEE (2021)
6. Cortes, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Transactions on robotics and Automation **20**(2), 243–255 (2004)
7. Fazli, P., Davoodi, A., Pasquier, P., Mackworth, A.K.: Complete and robust cooperative robot area coverage with limited range. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5577–5582. IEEE (2010)
8. Gabriely, Y., Rimon, E.: Spiral-stc: An on-line coverage algorithm of grid environments by a mobile robot. In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), vol. 1, pp. 954–960. IEEE (2002)
9. Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. Robotics and Autonomous systems **61**(12), 1258–1276 (2013)

10. Hazon, N., Mieli, F., Kaminka, G.A.: Towards robust on-line multi-robot coverage. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pp. 1710–1715. IEEE (2006)
11. Karapetyan, N., Benson, K., McKinney, C., Taslakian, P., Rekleitis, I.: Efficient multi-robot coverage of a known environment. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1846–1852. IEEE (2017)
12. Kong, C.S., Peng, N.A., Rekleitis, I.: Distributed coverage with multi-robot system. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pp. 2423–2429. IEEE (2006)
13. Oksanen, T., Visala, A.: Coverage path planning algorithms for agricultural field machines. Journal of field robotics **26**(8), 651–668 (2009)
14. Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E.: Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. Networks **72**(4), 411–458 (2018)
15. Rekleitis, I., Lee-Shue, V., New, A.P., Choset, H.: Limited communication, multi-robot team based coverage. In: IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, vol. 4, pp. 3462–3468. IEEE (2004)
16. Senthilkumar, K., Bharadwaj, K.K.: Multi-robot exploration and terrain coverage in an unknown environment. Robotics and Autonomous Systems **60**(1), 123–132 (2012)
17. Tang, J., Sun, C., Zhang, X.: Mstc: Multi-robot coverage path planning under physical constrain. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 2518–2524. IEEE (2021)
18. Tolstaya, E., Paulos, J., Kumar, V., Ribeiro, A.: Multi-robot coverage and exploration using spatial graph neural networks. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8944–8950. IEEE (2021)
19. Zheng, X., Jain, S., Koenig, S., Kempe, D.: Multi-robot forest coverage. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3852–3857. IEEE (2005)
20. Zhu, D., Tian, C., Sun, B., Luo, C.: Complete coverage path planning of autonomous underwater vehicle based on gbnn algorithm. Journal of Intelligent & Robotic Systems **94**(1), 237–249 (2019)