IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 10, 2020, accepted April 21, 2020, date of publication May 25, 2020, date of current version June 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2997095

Multi-Cleaning Robots Using Cleaning Distribution Method Based on Map Decomposition in Large Environments

XU MIAO, HYUN-SOON LEE, AND BO-YEONG KANG, (Member, IEEE)
School of Mechanical Engineering, Kyungpook National University, Daegu 41566, South Korea

Corresponding author: Bo-Yeong Kang (kby09@knu.ac.kr)

This work was supported by the National Research Foundation of Korea under Grant NRF-2019R1A2C1011270.

ABSTRACT Most cleaning robots have a good cleaning performance for small environments such as houses but require a longer cleaning time due to problems such as slow cleaning progress and low battery capacity, making the robots unsuitable for large environments such as libraries and airports. Cleaning large environments with multiple robots is faster than cleaning them with a single robot. Multi-cleaning robots can utilize several robots to simultaneously clean and share the task of cleaning among the robots. However, as the number of robots increases and the effective distribution of cleaning is not efficient during the cleaning process, the cleaning time will consequently be longer due to the frequent collisions between the robots. Therefore, to shorten the cleaning time, multi-cleaning robots require a coverage path planning that uses an effective cleaning distribution method in the cleaning process. In this paper, a coverage path planning using a cleaning distribution method based on map decomposition is proposed to reduce the cleaning time of multi-cleaning robots. The experimental results demonstrated that the proposed multi-cleaning robots' coverage path planning could be used in large environments in the presence of several types of obstacles. Furthermore, the cleaning time was found to be shorter than that of the previous methods in the case of multi-cleaning robots.

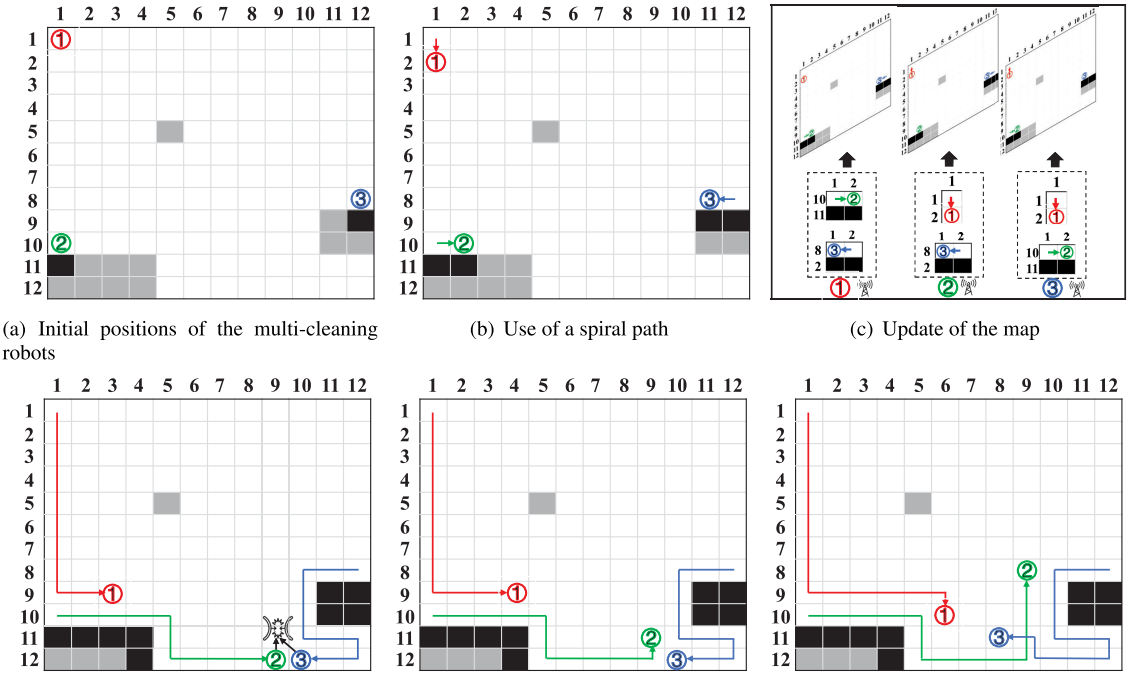
INDEX TERMS Coverage path planning, cleaning robot, computational efficiency, multi-robots system, robot path planning.

I. INTRODUCTION

Recently, multi-cleaning robots have become an important research topic in the field of cleaning robots [1]. Most cleaning robots are used in small environments such as houses; however, there is a growing need for robots that can be used in large environments such as libraries, airports, playgrounds, and warehouses [2]. When cleaning a sizable area using a single robot, the cleaning time is prolonged owing to the slow cleaning progress and the low battery capacity [3]. Therefore, when two or more robots are used, the robots can work simultaneously, thereby sharing the workload. Moreover, multi-cleaning robots are more robust [4], energy efficient [5], and demonstrate a superior spatial distribution [6] compared to using a single robot. Thus, in a sizeable environment, the cleaning time is shorter when multiple robots are used [4]. However, if the number of robots increases but the cleaning coordination does not perform effectively during the cleaning process, the cleaning time will be longer due to the frequent collisions between the robots [7]. To shorten the cleaning time, we described a multi-cleaning robots' coverage path planning (MRCPP) method using a cleaning distribution method for multi-cleaning robots. The MRCPP method calculates a path that can cover all the cleanable space while coordinating the multiple robots with a cleaning distribution method in a known or an unknown environment [8].

The process of the previous MRCPP method is summarized in Fig. 1 [9], [10]. The three cleaning robots (①, ②, and ③) perform cleaning along a spiral [11] or zigzag path [12] in Fig. 1(a). As shown in Fig. 1(b), each robot starts by moving

The associate editor coordinating the review of this manuscript and approving it for publication was Kathiravan Srinivasan.



(a) Initial positions of the multi-cleaning robots (b) Use of a spiral path (c) Update of the map

(d) Collision between cleaning robot ② and cleaning robot ③ (e) Use of the cleaning distribution method (f) Resolution of a collision

FIGURE 1. Summary of the previous MRCPP method.

on a spiral path. Each robot broadcasts its position to prevent collisions with the other robots during the cleaning process, as well as updates the complete map for the areas that have been cleaned, thereby avoiding redundant paths. In Fig. 1(c), three robots share the information of position, cleaned area, and detected obstacles, and update the shared map. If a robot completes the cleaning one location, it can move to the next uncleaned area. When moving to the next uncleaned area, the robot uses the shortest path planning method to determine the shortest traveling path.

When moving to the next uncleaned area, the robots may collide if they use the same travel path or try moving to the same uncleaned area [13]. Robot ② tries to clean the same area as robot ③, and both collide in Fig. 1(d). Therefore, multi-cleaning robots use the cleaning distribution method to solve this collision problem. As shown in Fig. 1(f), robot ② and robot ③ prevent a collision by using the cleaning distribution method shown in Fig. 1(e). Robot ③, which has resolved the collision, moves to the distributed uncleaned area by using the calculated the shortest traveling path. This process repeats until all the areas on the map are cleaned. If the map size is increased, the previous MRCPP methods encounter the following limitations.

1) As the multi-cleaning robots always update the information of the entire map to grasp the latest cleaning progress, the cleaning time increases.
2) The robots continuously check each other's position by means of a broadcast during the cleaning process, thereby utilizing the cleaning distribution method. Therefore, the cleaning time increases owing to the lack of independent cleaning ability.
3) When moving from a cleaned area to an uncleaned one during the cleaning process, the robots constantly check the location of the uncleaned area, thereby increasing the cleaning time.
4) In the previous research, the method for calculating the shortest travel path used by the multi-cleaning robots leads to an increase in the cleaning time as it considers all the grids of the grid map.

To address the limitations of the previous methods for a large environment, in this paper, an MRCPP method that uses an effective cleaning distribution technique based on map decomposition is proposed.

The proposed MRCPP method decomposes the entire map into sub-maps by using the map decomposition method [14] proposed by this paper's authors, eliminating the need to update the entire map. In the proposed method, each robot independently cleans only the distributed sub-map. As the robots are not required to communicate with each other by using the sub-map cleaning distribution, the multi-cleaning robots can clean the large environment more effectively. This method can also reduce the overall cleaning time using the corners and edges of the sub-maps instead of using the grid map when moving the uncleaned area and using the calculation method of the shortest path to travel.

The rest of this paper is organized as follows. Section II briefly surveys the previous methods, and Section III details the proposed MRCPP method. A comparison of the

performance of the previous methods and that of the proposed method is discussed in Section IV, and Section V presents the conclusion. (We recommend reading through the color-printed or PDF version of this paper because this paper provides pictures with various colors.)

II. RELATED WORK

The MRCPP methods are categorized as offline and online MRCPP methods [15], [16]. For the offline MRCPP methods [17], the environment is known in advance. Whereas, for the online MRCPP methods [18], the environment is unknown. This paper focuses on the online MRCPP methods. Many previous studies have analyzed map decomposition and multi-robot task allocation (MRTA) in online MRCPP methods. For example, Wagner *et al.* [19], [20] based an online MRCPP method without map decomposition on ant foraging. This method allows robots to share information and work together in a manner similar to the ants' communication with their pheromones. The advantage associated with the ant foraging method is that the map can be cleaned in cooperation with several robots without using a high-performance sensor. However, short cleaning times cannot be guaranteed because the robots are required to communicate when they are close together and cannot provide a global cleaning plan for the entire map. Research using MRCPP methods based on map decomposition has been conducted to improve the performance of the MRCPP method based on ant foraging. Most MRCPP methods based on map decomposition are classified into MRCPP methods [21]–[23] using boustrophedon decomposition [24], and MRCPP methods [9], [25]–[28] using a grid map.

MRCPP methods based on boustrophedon decomposition can be efficiently distributed to each cleaning robot by using a divide and conquer technique. Rekleitis *et al.* [21] proposed an MRCPP method of the team-based division of work using boustrophedon decomposition. In this MRCPP method, the multi-cleaning robots consist of an explorer and a coverer. The explorer explores the map and decomposes it into sub-maps by using boustrophedon decomposition, and the coverer proceeds to clean the decomposed sub-maps. In a team-based MRCPP method, robots interact using a line-of-sight communication method to decompose the map correctly. However, the line-of-sight communication method has a limited communication range. Thus, cleaning robots cannot efficiently clean when the communication range is exceeded, making it difficult to apply this method to large maps.

Kong *et al.* [22] and Rekleitis *et al.* [23] proposed a distributed MRCPP method using the boustrophedon decomposition method. Each cleaning robot uses a cycle algorithm [29] to clean up the decomposed sub-map. If the robot encounters an obstacle during cleaning, it can decompose the sub-map again. In this case, a process for integrating the decomposed maps is needed to prevent duplicate map decomposition. The integration of maps increases the communication among the robots and the total amount of communication data, adding to the cleaning time. Furthermore, in the case of integrated decomposed maps, without the information of the entire map, the robots cannot clean efficiently owing to the redundant paths [30].

In MRCPP methods using grid maps, the robots use the same map. The entire map indicating the location of the obstacles, the cleaned grid, and the uncleaned grid can be immediately confirmed using the state values of each grid. Therefore, the MRCPP methods using the grid map do not require a complicated map integration process as compared to the MRCPP methods using boustrophedon decomposition. Hazon *et al.* [25] and Senthilkumar and Bharadwaj [26] proposed an MRCPP method based on the spanning tree coverage (STC) method [31] using a grid map. The size of the grid map was set to twice the cleaning robot's diameter. The center positions of the grids were connected and set as a virtual wall. The virtual wall was represented with a tree structure, and all the grids were cleaned using the depth first search (DFS) method. The MRCPP method based on the STC method has the advantage of a robust and non-redundant path. However, if the environment is complicated or the initial positions of the robots are close to each other, there is no efficient cleaning distribution method, causing an imbalance in the amount of distributed cleaning to each robot [32], [33].

MRTA is required for an efficient cleaning distribution between the multi-cleaning robots and can be divided into centralized and decentralized approaches [34]. In the centralized approaches [35]–[37], each cleaning robot sends all the information to one central cleaning robot that subsequently allocates the cleaning to the other cleaning robots. One of the most significant shortcomings of the centralized approaches is a lack of robustness (i.e., failure of the entire system if the central cleaning robot fails). Conversely, in the decentralized approaches [38], [39], each cleaning robot shares information directly with other cleaning robots without a central robot that assigns the tasks. Therefore, decentralized approaches are resistant to local changes or failures if some robots malfunction [40]. Most decentralized approaches use a contract net protocol (CNP) method based on auctions [13]. In the CNP method, robots participate in the auction, and the most suitable robot performs the cleaning task [41].

In a previous CNP method, Gonzalez and Gerlein [9] provided a cooperative multi-agent system environment (BSA-CM), which is an MRCPP method based on the backtracking spiral algorithm (BSA) [42] using grid maps. In the BSA-CM method, each robot travels in a spiral path on the grid map. The uncleaned grid around the cleaned grid is defined as the backtracking point (BP). This method uses the collected BP to select the next uncleaned grid when the robot completes cleaning a grid. When selecting for the next uncleaned grid, an auction-based approach is used to distribute the task to the most suitable cleaning robot after surveying the amount of space each robot has cleaned. Yamachi *et al.* [27] proposed a method of integrating the grids to reduce the computational complexity when calculating the shortest path to the selected BP in the

BSA-CM method. Viet *et al.* [28] presented a BA* method that combines boustrophedon motions with the A* search algorithm to reduce the number of BPs in the BSA-CM method. Furthermore, the BoB [10] method, an MRCPP method based on the BA* method, was extended to a multi-cleaning robots version in [10]. When choosing a BP in the BoB method, the robots select the optimal BP using the greedy A* search (GA*) method. Moreover, to balance the distribution of tasks to each cleaning robot, the cleaning amount is allocated to the most suitable cleaning robot using the market-based approach after verifying the cleaning progress of each robot.

With respect to the other methods using the grid map, Kapanoglu *et al.* [43] developed the MRCPP method using the grid map combined with a genetic algorithm (GA). Sun *et al.* [44] and Zhu *et al.* [30] used a neural network based on a grid map in the MRCPP method. Gautam *et al.* [45] proposed the MRCPP method using the grid map combined with the cluster. The limitations of the previous MRCPP method based on the grid map are as follows:

- To prevent redundant paths, each robot updates the entire grid map by using shared cleaning information. Because of the frequent updates, the cleaning time increases.
- To avoid collisions, the robots check the positions of the others while cleaning each grid. Because of the excessive communication, the robots cannot work independently and take longer to clean the areas.
- Each robot must search all the uncleaned grids on the entire map to select the next uncleaned grid. Therefore, the cleaning time increases because of the number of additional calculations.
- When selecting the next uncleaned grid, the multi-cleaning robots calculate the path to travel to the next uncleaned grid by using the shortest path planning method. As the shortest path planning methods of the previous method calculate the path while exploring all the possible grids in the grid map, the cleaning time using the shortest path planning method increases with a larger grid map.

Due to the aforementioned limitations, the use of grid maps in large environments can significantly increase the cleaning time of multi-cleaning robots. In this paper, we propose an MRCPP method that can solve the limitations of the previous research and is efficient in large environments. First, as opposed to the previous studies using map decomposition, the proposed method does not require frequent updating of the map information as the robots only clean sub-maps that have been previously decomposed after proceeding with the map decomposition method [14] proposed by this paper's authors. Second, each robot has a distributed sub-map that can be cleaned separately by using a cleaning distribution method. Therefore, each robot can clean independently without sharing its progress. Third, when each robot travels from a cleaned grid to an uncleaned grid, it is not necessary to search the entire uncleaned grids because the proposed method only uses the corner grids of the sub-map. Finally, when the cleaning robot moves to the selected sub-map, the robot can travel quickly by using the shortest path planning method based on the edges and corners of the sub-map instead of the entire grid.

III. PROPOSED METHOD

In this paper, an MRCPP method that can effectively distribute cleaning areas to multi-cleaning robots in large environments is proposed. The conditions considered by the proposed method are as follows. First, the cleaning robots used in the experiment are homogeneous robots [13], and each robot performs map decomposition and map cleaning. Second, the size of one grid map (1×1) is set to the diameter of the robot, and the robot cleans one grid in one pass. All the grid maps were explored using same-sized robots. Finally, the grid map is represented as a two-dimensional Cartesian plane. Moreover, three algorithms were used to verify the accuracy of the cleaning algorithm through a simulation implementation because of the robot's hardware limitations (miscalculation in the moving distance due to an error by the attached sensor).

A. OVERVIEW OF THE PROPOSED METHOD

The overall flowchart of the proposed MRCPP method is shown in Fig. 2. First, the multi-cleaning robots are randomly classified as a sub-map decomposition robot and sub-map cleaning robots. Subsequently, the sub-map decomposition robot explores the entire map and decomposes the entire map into sub-maps by using the map decomposition method [14]. Moreover, the sub-map cleaning robots clean the decomposed sub-maps. Lastly, the sub-map decomposition robot and the sub-map cleaning robots complete the cleaning of the entire area through the cleaning distribution method.

The sub-map decomposition robot and the sub-map cleaning robots are defined in Equation (1).

$$\begin{aligned} R &= \{R_i | 1 \le i \le n\}, \quad |R_i| \ge 1 \\ R^M &= \{R^m | R^m \in R\}, \quad \left|R^M\right| = 1 \\ R^S &= \{R^s | R^s \in R, R^s \notin R^M\}, \quad \left|R^S\right| = n-1 \end{aligned} \tag{1}$$

Here, R refers to the set of n cleaning robots. Each robot R_i is a sub-map decomposition robot or a sub-map cleaning robot. R^M is the set of a sub-map decomposition robot. R^m means a sub-map decomposition robot, with $\left|R^M\right| = 1$ indicating that the sub-map decomposition robot is 1. R^S is the set of $n-1$ sub-map cleaning robots. R^s means a sub-map cleaning robot, and $\left|R^S\right| = n-1$ represents that there are $n-1$ sub-map cleaning robots. The robots are identified by their serial numbers, as i. In the following sections, the sub-map decomposition robot and the sub-map cleaning robot are represented as R^m and R^s, respectively.

R^m decomposes the entire map into sub-maps by using the map decomposition method [14]. In the decomposed sub-maps, each R^s selects an uncleaned sub-map. Equation (2)

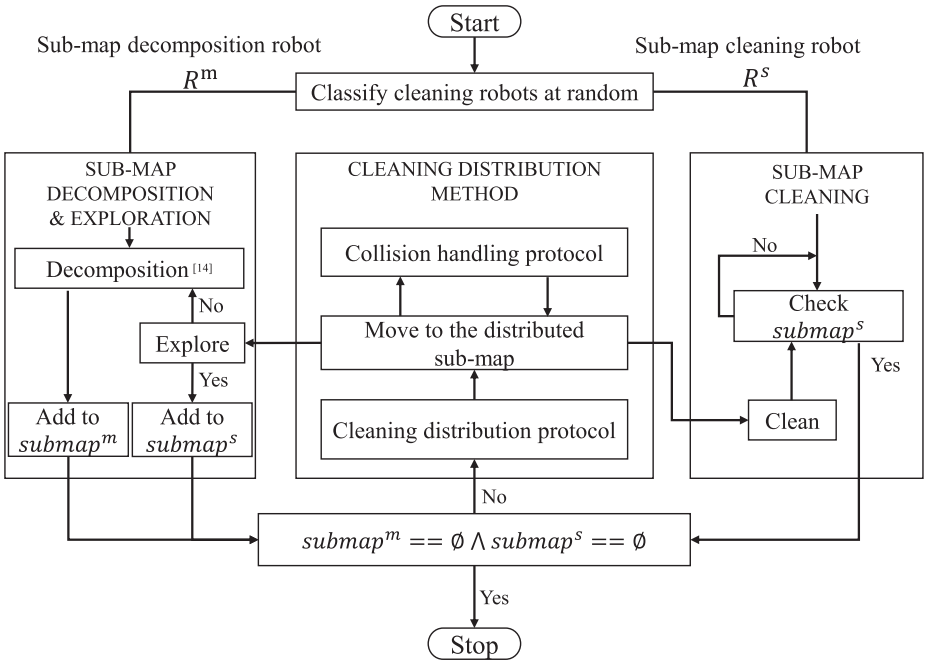


FIGURE 2. Proposed MRCPP method flow chart

shows the categorization between the sub-maps used by R^m and the sub-maps used by R^ss.

$$\begin{aligned} submap^m &= \{\ddot{x}_j | 1 \le j \le p\}, \ \left|\ddot{x}_j\right| \ge 1 \\ submap^s &= \{\ddot{x}_j | \ddot{x}_j \in submap^m\} \end{aligned} \tag{2}$$

where, $submap^m$ is the sub-map set used by R^m, and $submap^s$ indicates the sub-map set used by R^s. In addition, $\ddot{x}$ means a sub-map. p is the number of sub-maps.

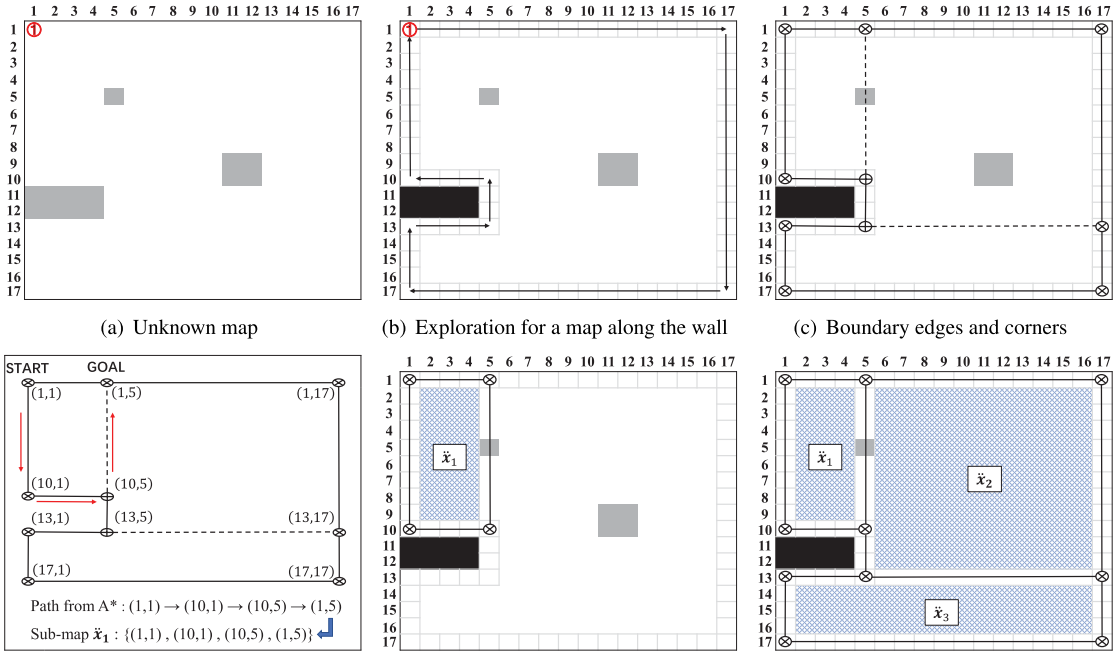
The process of cleaning using R^m and R^s can be described in detail as follows. In Fig. 2, R^m decomposes the entire map into sub-maps through the map decomposition method and subsequently adds the decomposed sub-maps to the $submap^m$. R^m also uses a cleaning distribution method to explore for the sub-maps and confirms that there are sub-maps in both $submap^m$ and $submap^s$ before using the cleaning distribution method. Consequently, if there are sub-maps in both $submap^m$ and $submap^s$, R^m employs the cleaning distribution method. When using the cleaning distribution method, R^m selects the sub-map from $submap^m$ by using the cleaning-distribution protocol and then R^m travels to the selected sub-map. When robots collide during traveling, the collision handling protocol is applied. When R^m arrives at the distributed sub-map, it begins to explore the area. If an obstacle is detected during this process, R^m uses the map decomposition method again. However, if there are no detected obstacles during the exploration, R^m adds this sub-map to the $submap^s$ and then employs the cleaning distribution method again.

Each R^s determines if the sub-map is included in $submap^s$, and R^s uses the cleaning distribution method to select an uncleaned sub-map. R^s, like R^m, confirms that there are sub-maps in both $submap^m$ and $submap^s$ before using the cleaning distribution method. Consequently, if there are sub-maps in both $submap^m$ and $submap^s$, R^m use the cleaning distribution method. R^s selects a sub-map from $submap^s$ by using the cleaning-distribution protocol and travels to the selected sub-map. When robots collide during traveling, the collision handling protocol is applied. When R^s arrives at the distributed sub-map, it begins to clean. After completing the process, R^s checks $submap^s$ again. If both $submap^m$ and $submap^s$ are the empty set (ϕ), the robots have finished cleaning the area, and the MRCPP method is stopped. The pseudo-code of the proposed method is shown in Algorithm1.

B. SUB-MAP DECOMPOSITION AND EXPLORATION

R^m decomposes a given map into sub-maps and explores the sub-maps in the following steps. R^m decomposes the map into sub-maps by using the map decomposition method [14] proposed by this paper's authors. The process [14] to decompose an entire map into multiple rectangles is shown in Fig. 3. Robot ① is R^m. The gray grid is an obstacle before detection, and the black grid denotes an obstacle that has been detected. The signs ⊗ and ⊕ are defined as concave and convex corners, respectively. Solid lines indicate the boundary edges, and dotted lines are the decomposition edges. Finally, the solid arrow indicates the traveling path of the cleaning robot.

Fig. 3(a) represents the unknown map, and the position of R^m (①) is the coordinate set $(1, 1)$. R^m uses an infrared sensor to detect the outline of the map along the wall in Fig. 3(b). During the detection of the outline of the map using R^m, the boundary edges and the corners



(a) Unknown map (b) Exploration for a map along the wall (c) Boundary edges and corners

(d) Decomposition of sub-maps using A* method (e) Sub-map $\ddot{x}_1$ (f) Map decomposition into three sub-maps

FIGURE 3. Procedural process for the decomposition of sub-maps using the previously proposed map decomposition method [14].

Algorithm 1 Proposed Method

Initialize:
1) Randomly select one robot as R^m among the multi-cleaning robots, and the other as the R^s.
2) R^m follows the walls and saves information about the corners and edges of the unknown map.
3) $submap^m \leftarrow R^m$ use **Sub-map decomposition and exploration**.
$submap^s \leftarrow \phi$, all $R_{i\_state} \leftarrow \phi$, $\{R_{i\_state} = \phi$: cleaning robot's idle state; $R_{i\_state} = \ddot{x}_j$: cleaning robot's activity state;$\}$

while $submap^s \neq \phi \vee submap^m \neq \phi$ **do**
  for all $R_i \in R$ **do**
    if $R_{i\_state} == \phi$ **then**
      Execute **Cleaning distribution**;
    else
      if $R_i == R^m$ **then**
        Execute **Sub-map decomposition and exploration**;
      else
        Execute **Sub-map cleaning**;
      end if
    end if
  end for
end while

are collected, and some obstacles are detected. The collected boundary edges (solid lines, −), concave corners (⊗), and convex corners (⊕) are displayed in Fig. 3(c).

R^m uses two convex corners of the collected corners to decompose as the decomposed edge (dotted line, −−) $\langle(1, 5), (10, 5)\rangle$ and the decomposed edge $\langle(13, 5), (13, 17)\rangle$. Consequently, R^m collects the information for 12 edges and 10 corners as shown in Fig. 3(d). The 12 edges are at $\langle(1, 1), (1, 5)\rangle$, $\langle(1, 5), (1, 17)\rangle$, $\langle(1, 17), (13, 17)\rangle$, $\langle(13, 17), (17, 17)\rangle$, $\langle(17, 17), (17, 1)\rangle$, $\langle(17, 1), (13, 1)\rangle$, $\langle(13, 1), (13, 5)\rangle$, $\langle(13, 5), (10, 5)\rangle$, $\langle(10, 5), (10, 1)\rangle$, $\langle(10, 1), (1, 1)\rangle$, $\langle(1, 5), (10, 5)\rangle$, $\langle(13, 5), (13, 17)\rangle$, and the 10 corners are at (1, 1), (1, 5), (1, 17), (13, 17), (17, 17), (17, 1), (13, 1), (13, 5), (10, 5), (10, 1).

R^m uses the A* method [46] to decompose an unknown map into sub-maps based on the collected edges and corners, and randomly chooses one of the edges from the collected edges for the sub-map decomposition. Next, the two corners connected with the selected edge are set as the starting point and the target point. For example, when R^m selects edge $\langle(1, 1), (1, 5)\rangle$ in Fig. 3(d), the starting point and the target point are set to (1, 1) and (1, 5), respectively. R^m also explores for edges connected from the starting point to the target point between the other edges excluding the edge selected by the A* method. The explored edges are $\langle(1, 1), (10, 1)\rangle$, $\langle(10, 1), (10, 5)\rangle$, $\langle(10, 5), (1, 5)\rangle$, as indicated by the arrows. Sub-map $\ddot{x}_1$ consists of the three explored edges and the first selected edge $\langle(1, 1), (1, 5)\rangle$ in Fig. 3(e). This is represented as the blue region. This process is repeated and is stopped when the sub-map is decomposed for all the edges. Thus, Fig. 3(f) shows that the A* method was used to decompose the entire map into three sub-maps (defined as $\ddot{x}_1$, $\ddot{x}_2$, and $\ddot{x}_3$). Moreover, three sub-maps are saved in $submap^m$ used by R^m.

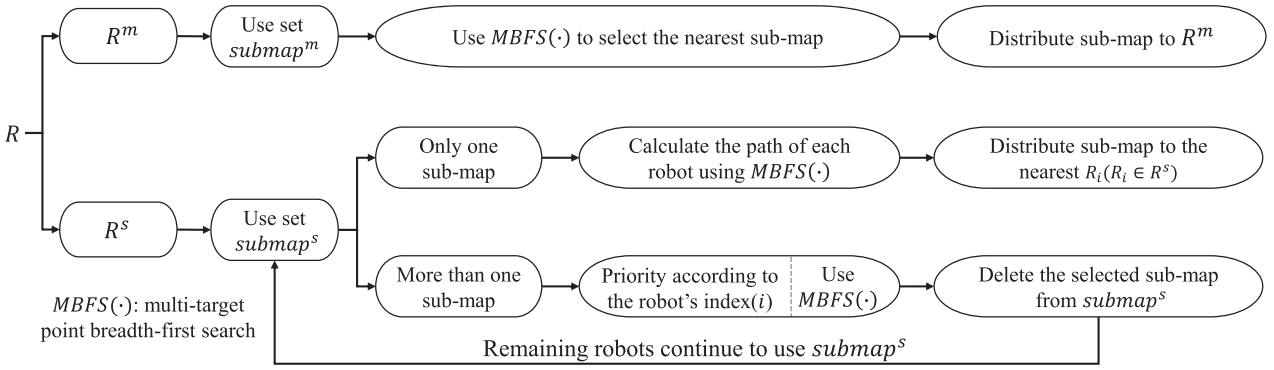


FIGURE 4. Cleaning-distribution protocol

For using the sub-map distributed to each R^s, R^m explores the three decomposed sub-maps and travels along the boundary edge of the sub-map in exploration. If an obstacle is detected while traveling, the sub-map is decomposed again. Whereas, if R^m does not detect an obstacle, this sub-map is saved in $submap^s$. The pseudo-code for all the processes is presented in Algorithm2.

Algorithm 2 Sub-Map Decomposition and Exploration

Input: $submap^m$, $submap^s$, R^m
Output: $submap^m$, $submap^s$, R^m
if $R^m_{i\_state} == \phi$ **then**
  $submap^m \leftarrow$ Execute **Map decomposition** [14];
else
  $O_{boundary} \leftarrow$ Move along $R^m_{i\_state}$; {Sub-map explore}
  if $O_{boundary} == \phi$ **then**
    $submap^s \leftarrow submap^s \cup \{R^m_{i\_state}\}$
  else
    $submap^{new}_{set} \leftarrow$ Execute **Map decomposition**;
    $submap^m \leftarrow submap^m \cup submap^{new}_{set}$;
  end if
  $submap^m \leftarrow submap^m - \{R^m_{i\_state}\}$;
  if $submap^m == \phi$ **then**
    Cancel R^m's permission and R^m change to R^s;
  end if
end if

C. CLEANING DISTRIBUTION METHOD

A cleaning distribution method for efficient sub-map selection in $submap^m$ or $submap^s$ is proposed for each robot. The proposed cleaning distribution method consists of a cleaning-distribution protocol and a collision-handling protocol. The cleaning-distribution protocol computes the path and selects a sub-map. The collision-handling protocol manages collisions that occur while the robots travel to the sub-maps.

1) CLEANING-DISTRIBUTION PROTOCOL

In the proposed cleaning-distribution protocol, shown in Fig. 4, R^m selects the closest sub-map from $submap^m$ by applying the multi-target point breadth-first search (MBFS) method. The MBFS method is the shortest path planning method based on the breadth-first search (BFS) method [47] for multi-target points using the information of the edges and the corners for the sub-map.

Each R^m selects a sub-map from $submap^s$. When there is only one sub-map in $submap^s$, the R^ss compare the Euclidean distances by using the MBFS method. Consequently, the closest robot selects the sub-map. Whereas, for multiple maps in $submap^s$, the R^ss use the MBFS method in the given numerical order. The R^s with the smallest number uses the MBFS method to select the closest sub-map with the closest distance. The remaining R^ss use the cleaning distribution-protocol to select the closest sub-maps from $submap^s$ in turn.

2) COLLISION-HANDLING PROTOCOL

After using the cleaning-distribution protocol, the robots travel the selected sub-maps. A collision occurs while traveling to the sub-maps because each robot uses the same information of the edges and the corners on the map. The proposed method uses a collision-handling protocol when a collision occurs between robots. This protocol has three cases between two robots, as shown in Fig. 5. In this figure, we need to decompose the map into sub-maps in advance; therefore, we set R^m to a high priority.

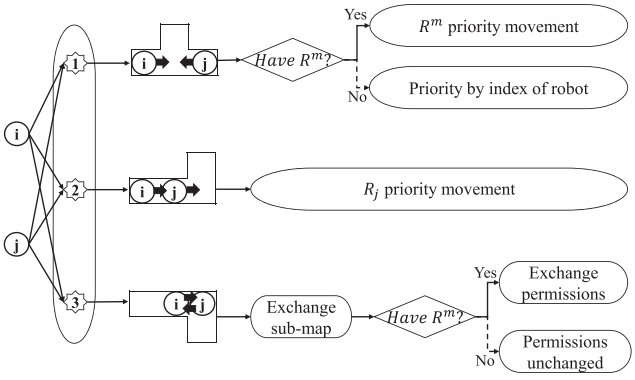
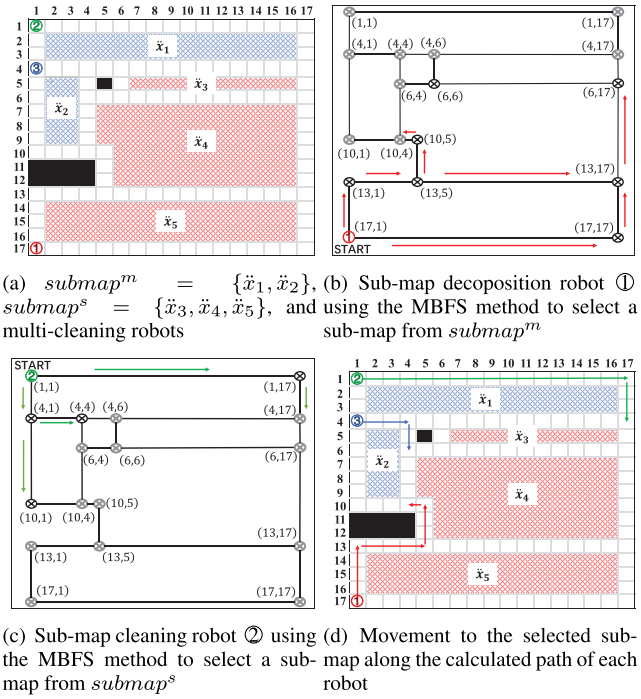


FIGURE 5. Collision-handling protocol

In the collision-handling protocol, collision-processing I occurs because robot R_i and robot R_j are trying to move to the

same position. To resolve this conflict, either R_i or R_j is identified as R^m. R^m travels as the highest priority to decompose the map. Whereas, the robots travel in a sequential order without an R^m. In collision-processing II, when a robot travels to an arbitrary position, a collision occurs because another robot is already in that position. For example, a collision occurs when R_i tries to travel to the position of R_j. To resolve this conflict, R_j moves to another position, subsequently R_i travels to the original position of R_j. In collision-processing III, the collision occurs because R_i and R_j try to travel in opposite positions. In this case, R_i and R_j exchange a sub-map with each other for efficient cleaning. Moreover, R_i and R_j travel to the exchanged sub-map. It is necessary to exchange the permissions of the robots as well, as R^m and R^s have different handling processes for the sub-map. Therefore, it is determined that either R_i or R_j is R^m. If either is R^m, R^m and R^s exchange permissions.



(a) $submap^m = \{\ddot{x}_1, \ddot{x}_2\}$, $submap^s = \{\ddot{x}_3, \ddot{x}_4, \ddot{x}_5\}$, and multi-cleaning robots

(b) Sub-map decoposition robot ① using the MBFS method to select a sub-map from $submap^m$

(c) Sub-map cleaning robot ② using the MBFS method to select a sub-map from $submap^s$

(d) Movement to the selected sub-map along the calculated path of each robot

FIGURE 6. Process of the cleaning distribution method in the proposed MRCPP method.

3) DESCRIPTION OF THE CLEANING DISTRIBUTION METHOD

The entire process using the proposed cleaning distribution method (Fig. 2) is shown in Fig. 6. Fig. 6(a) shows three robots, $submap^m$, and $submap^s$. Robot ① is set as R^m, and the coordinates of robot ① are (17, 1). Robot ② and robot ③ are set to R^s, and their coordinates are (1,1) and (4,1), respectively. There are two sub-maps, $\ddot{x}_1$ and $\ddot{x}_2$, in $submap^m$ that R^m uses. The R^ss use three sub-maps ($\ddot{x}_3$, $\ddot{x}_4$, and $\ddot{x}_5$) in $submap^s$. Information about the edges and the corners of the sub-maps is represented in Fig. 6(b). The edges of the 22 sub-maps are $\langle(1, 1), (4, 1)\rangle$, $\langle(4, 1), (4, 4)\rangle$, $\langle(4, 4), (4, 6)\rangle$, …, $\langle(17, 1), (17, 17)\rangle$, $\langle(17, 17), (13, 17)\rangle$, $\langle(13, 5), (13, 1)\rangle$, and the corners of these sub-maps are shown as (1, 1), (1, 17), (4, 1), …, (13, 17), (17, 1), (17, 17).

The three robots use the cleaning distribution method to select the sub-maps. Robot ① is R^m; therefore, the MBFS method is used according to the cleaning-distribution protocol. Fig. 6(b) shows that robot ① (R^m) uses the cleaning-distribution protocol, as shown in Fig. 6(b). First, the corners of sub-map $\ddot{x}_1$ and sub-map $\ddot{x}_2$ of the $submap^m$ are set as the target points and are indicated in gray. Second, the target point of the shortest distance is explored using the BFS method based on the edge information for all the sub-maps. Therefore, R^m starts exploring the traveling path at the coordinates (17, 1) and selects the edge connected with (17, 1). The edges connected to the coordinates (17, 1) are the two edges of $\langle(17, 1), (13, 1)\rangle$ and $\langle(17, 1), (17, 17)\rangle$. The connected corners (13, 1) and (17, 17) are not gray, and therefore the robot continues to search the edge. Thus, the edge connected to the corner (13, 1) is $\langle(13, 1), (13, 5)\rangle$.

For the connected edge, the corner (13, 5) is not gray, and thus, R^m explores the edges connected to the corner (17, 17). The edge connected to the corner (17, 17) is edge $\langle(17, 17), (13, 17)\rangle$. For this edge, the edges connected to (13, 5) and (6, 17) are continuously found as the corner (13, 17) is not gray. Because of repeating the previous process, the edge $\langle(10, 5), (10, 4)\rangle$ connected to the corner (10, 5) is found. The found paths are indicated by solid lines with an arrow. Third, if one of the target points is a detected target point, the MBFS method is stopped. Therefore, as the gray corners (10, 4) of the target points are detected, the MBFS method is stopped. R^m selects sub-map $\ddot{x}_2$ because (10, 4) (the detected gray corners) are the corners of sub-map $\ddot{x}_2$. Thus, the traveling path calculated by the MBFS method is {(17, 1), (13, 1), (13, 5), (10, 5), (10, 4)}.

As robot ② (R^s) and robot ③ (R^s) have three sub-maps in $submap^s$ according to the cleaning-distribution protocol, the MBFS method is used in the numerical order of R^s. Therefore, Fig. 6(c) shows how robot ② selects a sub-map from $submap^s$ by using the MBFS method. Robot ② selects sub-map $\ddot{x}_3$ from $submap^s$. Consequently, the traveling path calculated by the MBFS method is {(1, 1), (1, 17), (4, 17)}. As $\ddot{x}_3$ has already been selected, robot ③ uses the MBFS method to select either sub-map $\ddot{x}_4$ or sub-map $\ddot{x}_5$. Therefore, robot ③ selects sub-map $\ddot{x}_4$. Finally, in Fig. 6(d), each robot travels to the selected sub-map. The pseudo-code of the proposed cleaning distribution method is presented in Algorithm3.

D. SUB-MAP CLEANING

In this section, we describe how to clean a sub-map by using the potential method [12]. In this method, potential values are set for each grid of the map. The robot travels from the highest potential value to the lowest value, thereby cleaning the grid while creating a spiral path. The process of cleaning using the spiral path of the proposed potential method is shown in Fig. 7. In Fig. 7(a), R_i means R^s, and the position

Algorithm 3 Cleaning Distribution Method

Input: $submap^m$, $submap^s$, R
Output: $submap^m$, $submap^s$, R
if $R == R^m$ **then**
  $(R_i^m, submap^m) \leftarrow$ Use cleaning-distribution protocol;
end if
if $R == R^s$ **then**
  $(R_i^s, submap^s) \leftarrow$ Use cleaning-distribution protocol;
end if
{↓ Follow each own path}
while Follow along path of R_i **do**
  Check the positions of other robots
  if Conflict **then**
    Use collision-handling protocol;
  else
    Follow along path of R_i;
  end if
end while

of R_i is defined as the coordinates (13,1). R_i begins to clean sub-map $\ddot{x}_5$. As R^s has collected information on the boundary edges and the corners of sub-map $\ddot{x}_5$, the local grid map is based on sub-map $\ddot{x}_5$ represented in Fig. 7(b).

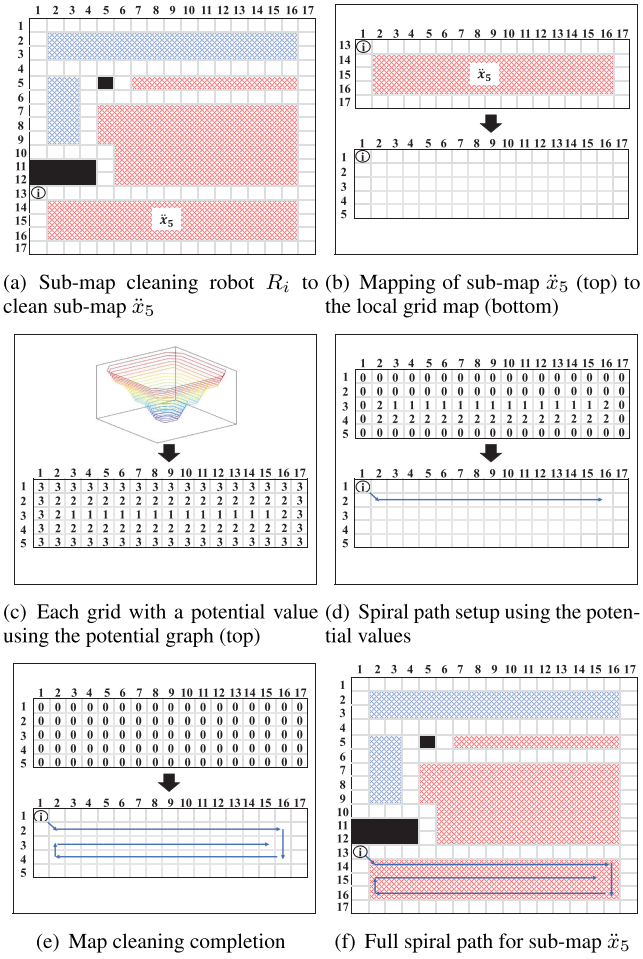
Fig. 7(c) shows that each potential value is set for all the grids by using a potential graph. In the exploration of sub-map $\ddot{x}_5$ along the wall, R^m has already cleaned the grids at the boundary edges. Therefore, the potential values of the boundary edges have been changed from three (see Fig. 7(c)) to zero in Fig. 7(d). This figure shows the cleaning process of R_i, and the path is represented as the blue line. The path marked with the blue line also has the potential values of two to zero.

Whenever R^s travels, the potential values are checked for the three (up, left, and right) grids connected with a grid of R^s. The robot moves to the grid with the largest potential value. The above processes are then repeated until the potential values of all the grids on the map become zero. In Fig. 7(e), all the grids are cleaned by the robot, and the potential values are changed to zero by using the potential graph. Thus, cleaning is completed for sub-map $\ddot{x}_5$. Moreover, Fig. 7(f) shows that, the entire spiral path created by R^s for sub-map $\ddot{x}_5$. The proposed pseudo-code for sub-map cleaning is given in Algorithm4.

IV. EXPERIMENTAL RESULTS

A simulation was conducted to prove the performance of the proposed MRCPP method by using a parallel computing toolbox provided in MatLab (R2017a). All the experiments were conducted on a Windows operating system with a specification of 3.60-GHz Intel Core i7-7700 and memory of 8.00 GB. The cleaning robot used in the experiment was the kobuki robot[1] developed by Yujin Robotics (Korea). The kobuki robot has a diameter of 35.15 cm and a travel speed

[1]Functional specification of kobuki: http://kobuki.yujinrobot.com/about2/



(a) Sub-map cleaning robot R_i to clean sub-map $\ddot{x}_5$
(b) Mapping of sub-map $\ddot{x}_5$ (top) to the local grid map (bottom)
(c) Each grid with a potential value using the potential graph (top)
(d) Spiral path setup using the potential values
(e) Map cleaning completion
(f) Full spiral path for sub-map $\ddot{x}_5$

FIGURE 7. Cleaning of a selected sub-map using the spiral path with the potential values.

of 70 cm/s. The number of multi-cleaning robots was set to 2-4. The performance of the BSA-CM [9], [48], the BoB method [10], and the proposed MRCPP method was compared using a set number of multi-cleaning robots.

A. CLEANING PERFORMANCE COMPARISON

Based on Lee *et al.* [49]' Fractal World Generator, six maps of size 100×100 with obstacles of various sizes and numbers were created, as shown in Fig. 8. The size of the 100 × 100 grid was increased 1 to 5 times to compare the cleaning performance for a larger map. Here, the shapes of the obstacles were maintained for each of the six maps. Thus, the map size for each map was 100×100 to 500×500, and the total number of maps used in the experiment was 30.

The coverage ratio and the cleaning time were used to compare the cleaning performance of the two previous methods and the proposed method. The coverage ratio was calculated as the ratio of the cleaned grids to the empty grids in a grid map for a given environment. Therefore, when the multi-cleaning robots cleaned all the empty grids in the grid map, it indicated that 100% of the cleaning had been completed, implying a complete coverage ratio, expressed

Algorithm 4 Sub-Map Cleaning

Input: R^s
Output: R^s
Initialize: Create a grid map using R_{i_state}, and set the value for each *grid* according to the potential graph;
while true **do**
  Calculate a spiral path based on the potential values to execute cleaning;
  if blind alley occurs in *grid* **then**
    if $\exists grid > 0$ **then**
      $grid_{uncleaned} \leftarrow$ Select an uncleaned grid;
      $path \leftarrow$ Execute **A***($grid_{uncleaned}$, grid map);
      Move along the *path* to the $grid_{uncleaned}$;
    else
      Break;
    end if
  end if
end while

as 1.0. Table 1 shows the results of the coverage ratios for maps using the three MRCPP methods based on 2-4 robots. For the proposed six maps, all the three methods showed a complete coverage ratio of 1.0.

The cleaning time was compared for the two previous methods and the proposed method. Equation (3) was used to calculate the cleaning time.

$$Clean_{time} = Algorithm_{execution} + Travel_{time} \tag{3}$$

Here, $Clean_{time}$ shows the total time taken to clean, $Algorithm_{execution}$ means the time taken to execute the algorithm, and $Travel_{time}$ is the time taken for the cleaning robot to travel. $Travel_{time}$ is added to $Clean_{time}$ to represent the actual cleaning time. $Travel_{time}$ is calculated using Equation (4).

$$Travel_{time} = \rho \times \max_{1 \le i \le n} Grid\ (R_i) \tag{4}$$

In Equation (4), ρ means the time used when the cleaning robot (R_i) travels a grid. Grid is the number of grids, and n is the number of robots. The value of ρ (0.5 s) was calculated based on the kobuki robot's diameter of 35.15 cm and traveling speed of 70 cm/s. When calculating $Travel_{time}$, the robots used the number of grids that were cleaned the most by the robots in the result of cleaning at the same time.

Fig. 9 shows the cleaning times calculated for the BSA-CM method, the BOB method, and the proposed method using two multi-cleaning robots. In Fig. 9, the x-axis is the map size from 100×100 to 500×500, and the y-axis is the time it takes to clean the area. The experimental results showed that the proposed method reduced the cleaning time significantly as the size of the six maps increased as compared to the conventional BSA-CM method and the BoB method.

Fig. 9(c) shows that the cleaning time of the proposed method was less than the time spent for the BSA-CM method by 14 min 45 s, 27 min 17 s, 1 h 43 min 26 s, and 2 h 33 min 28 s in map sizes ranging from 200 × 200 to 500 × 500, respectively. Moreover, the cleaning time of the proposed method was faster than the cleaning time of the BoB method by 3 min 51 s, 18 min 14 s, 57 min 22 s, and 1 h 46 min 27 s, respectively. In particular, for the size of 500 × 500, the cleaning time of the proposed method was much less than the times of the other methods. An analysis of the proposed method and the previous methods for the six maps showed that the average cleaning time of the proposed method was 2 h 33 min 33 s shorter than that of the BSA-CM method and 1 h 37 min 31 s less than the cleaning time of the BoB method.

In Fig. 9, it is shown that the proposed method significantly reduces the cleaning time when the map size is 500 × 500 for the six maps. Therefore, in Table 2, we compare the cleaning time for each method according to the change in the number of cleaning robots for the six maps with a map size of 500 × 500. The number of cleaning robots set for the comparison of cleaning times was changed from two to four. Table 2 shows that the cleaning time of the proposed method

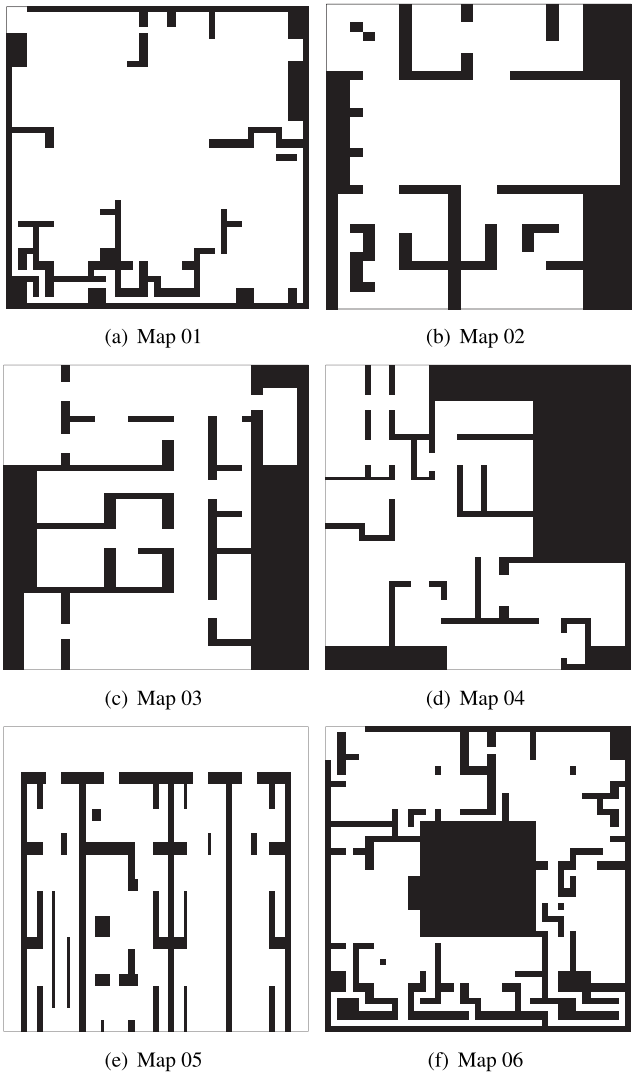


FIGURE 8. Maps used for the experiment (size of each map: 100 × 100 ~ 500 × 500, Interval: 100 × 100).

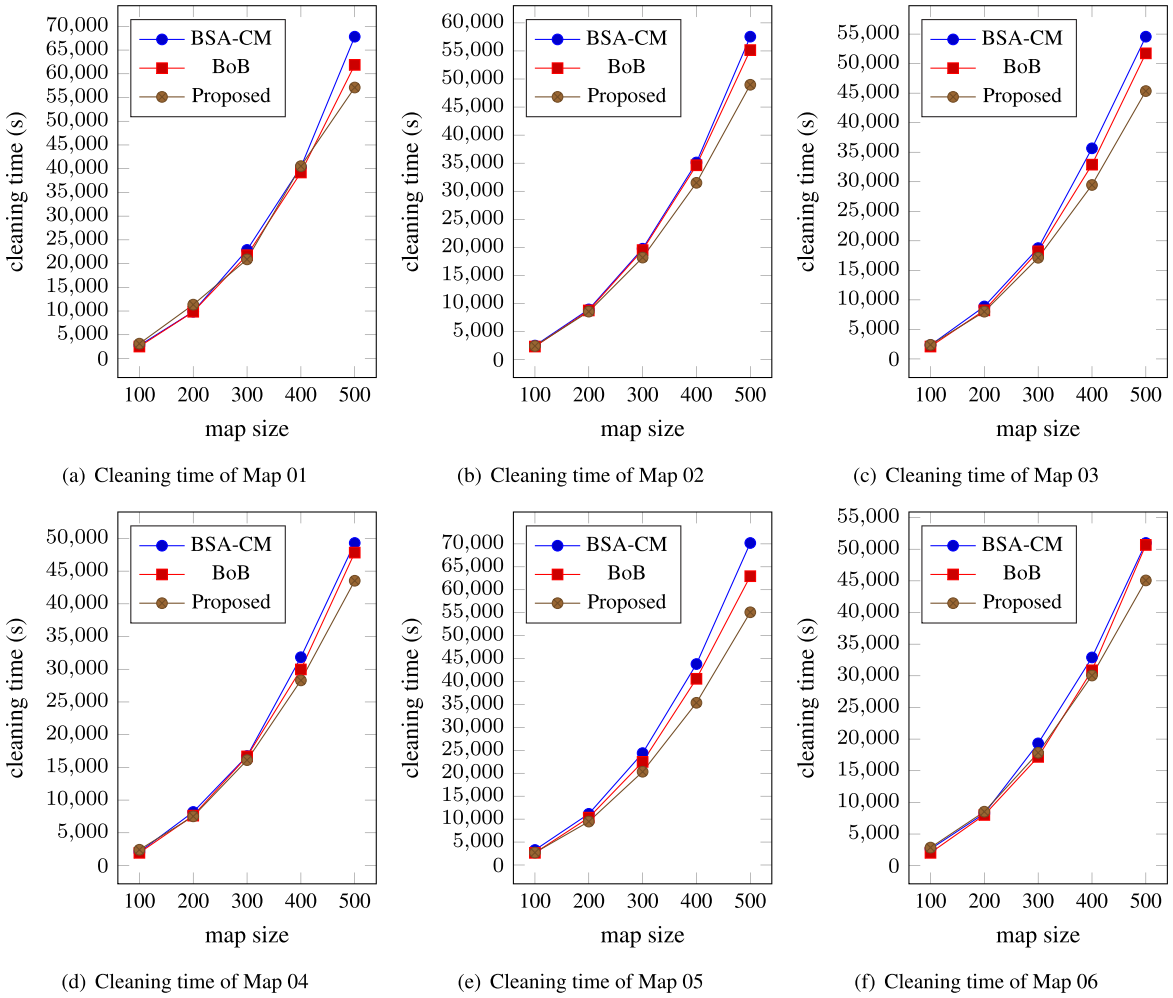


FIGURE 9. Comparison of the cleaning time between the proposed method and two previous methods for six maps of different sizes by using two cleaning robots (x-axis: map size, y-axis: cleaning time (unit: time (s))).

TABLE 1. Coverage ratio for maps cleaned by multi-cleaning robots.

Robots	Map	Size	BSA-CM	BoB	Proposed
2	Map 01	100 × 100	1.0	1.0	1.0
	Map 02	100 × 100	1.0	1.0	1.0
	Map 03	100 × 100	1.0	1.0	1.0
	Map 04	100 × 100	1.0	1.0	1.0
	Map 05	100 × 100	1.0	1.0	1.0
	Map 06	100 × 100	1.0	1.0	1.0
3	Map 01	100 × 100	1.0	1.0	1.0
	Map 02	100 × 100	1.0	1.0	1.0
	Map 03	100 × 100	1.0	1.0	1.0
	Map 04	100 × 100	1.0	1.0	1.0
	Map 05	100 × 100	1.0	1.0	1.0
	Map 06	100 × 100	1.0	1.0	1.0
4	Map 01	100 × 100	1.0	1.0	1.0
	Map 02	100 × 100	1.0	1.0	1.0
	Map 03	100 × 100	1.0	1.0	1.0
	Map 04	100 × 100	1.0	1.0	1.0
	Map 05	100 × 100	1.0	1.0	1.0
	Map 06	100 × 100	1.0	1.0	1.0

is shorter according to the increased number of robots than the times of the previous methods. For example, when two robots were used, the average cleaning time for the proposed method for the six maps was 13 h 34 min13 s. The average cleaning time of the BSA-CM method and the average cleaning time of the BoB method were16 h 13 min 20 s and 15 h 17 min 17 s, respectively. Therefore, the average cleaning time of the proposed method was shorter by 2 h 39 min 07 s than that of the BSA-CM method and was 1 h 43 min 04 s less than that of the BoB method.

For the proposed method, the average cleaning time for the six maps was 9 h 30 min 52 s when using three cleaning robots. The average cleaning time in the BSA-CM method was 11 h 36 min 51 s, and the average cleaning time in the BoB method was 11 h 11 min 34 s. Thus, the average cleaning time in the proposed method was faster by 2 h 05 min 59 s than that in the BSA-CM method and by 1 h 40 min 42 s than the time for the BoB method. With four robots in six maps, the average cleaning time used was 7 h 22 min 50 s for the proposed method, 9 h 49 min 58 s for the BSA-CM method, and 8 h 33 min 11 s for the BoB method. Consequently, the proposed method decreased the average cleaning time by 2 h 27 min 08 s as compared to the BSA-CM

TABLE 2. Results of the cleaning time for three MRCPP methods using multi-cleaning robots in six 500 × 500 maps (unit: seconds (s), time format: hours (h): minutes (min): seconds (s)).

Robots	Map (size: 500 × 500)	BSA-CM	BoB	Proposed
2	Map 01	67,818	61,873	**57,106**
	Map 02	57,541	55,153	**48,973**
	Map 03	54,560	51,739	**43,351**
	Map 04	49,302	47,833	**43,528**
	Map 05	70,193	62,955	**55,091**
	Map 06	50,984	50,668	**45,067**
	Average	58,400	55,073	**48,853**
	(time)	(16h:13min:20s)	(15h:17min:17s)	**(13h:34min:13s)**
3	Map 01	49,950	43,210	**38,490**
	Map 02	39,904	39,212	**35,252**
	Map 03	38,854	36,315	**33,672**
	Map 04	34,818	33,320	**30,011**
	Map 05	50,639	51,101	**38,183**
	Map 06	36,701	38,607	**29,903**
	Average	41,811	40,294	**34,252**
	(time)	(11h:36min:51s)	(11h:11min:34s)	**(9h:30min:52s)**
4	Map 01	44,905	34,139	**29,884**
	Map 02	31,639	30,589	**28,058**
	Map 03	33,494	30,766	**23,437**
	Map 04	28,304	26,408	**22,173**
	Map 05	43,130	35,864	**30,387**
	Map 06	30,919	26,978	**25,481**
	Average	35,399	30,791	**26,570**
	(time)	(9h:49min:58s)	(8h:33min:11s)	**(7h:22min:50s)**

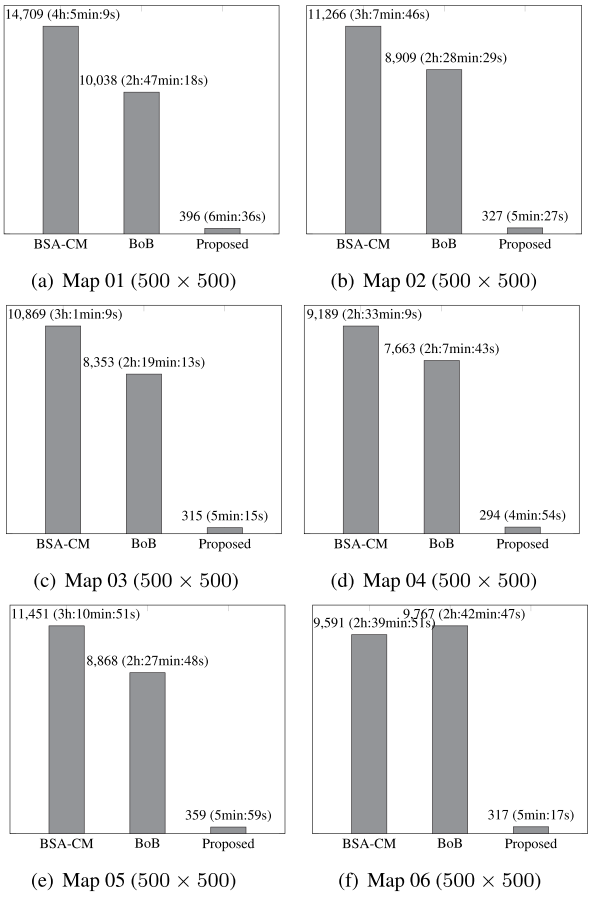


FIGURE 10. Comparison of the execution time of the algorithm in three MRCPP methods using two cleaning robots (unit: seconds(s), time format: hours (h): minutes (min): seconds (s)).

method and by 1 h 10 min 21 s as compared to the BoB method.

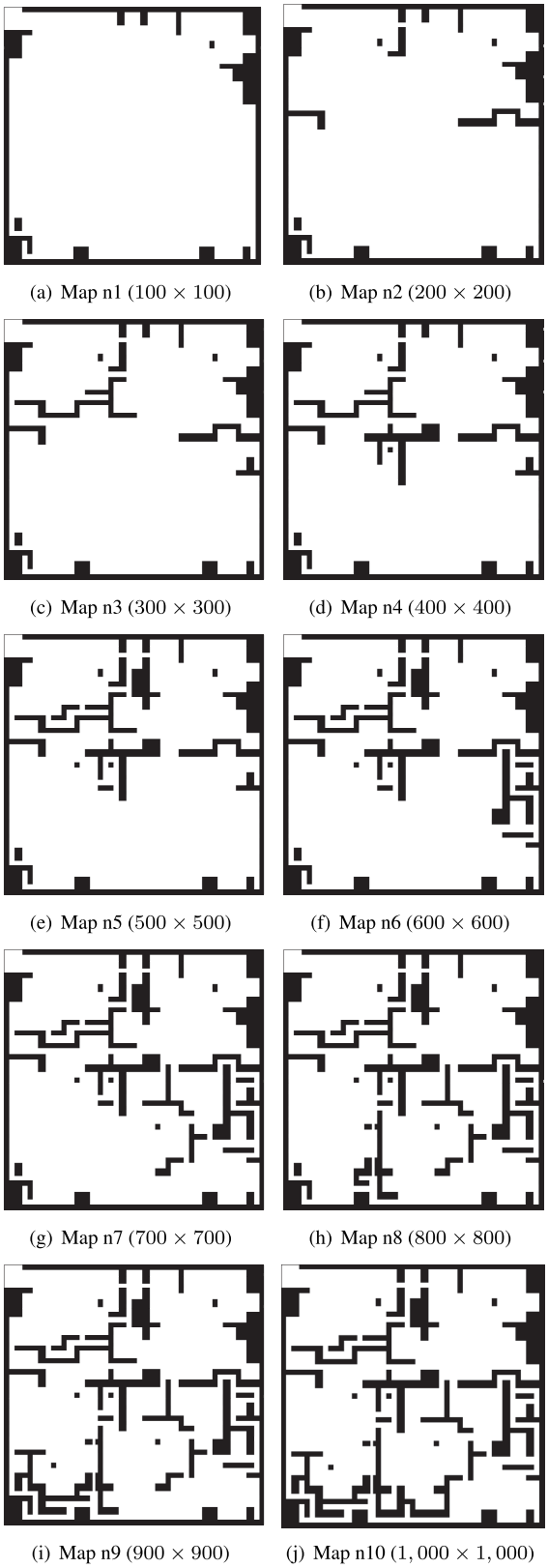
As the proposed method used a shorter algorithm execution time than the previous methods, the total cleaning time was decreased. Fig. 10 shows the results of the algorithm execution time for three methods with two cleaning robots in the six maps of 500 × 500. In the figure, the algorithm execution time is indicated by the gray bars. Fig. 10 shows that the proposed method used less time for algorithm execution than the other two methods for all the experiments. For example, the algorithm execution time for each method is shown for map 04 in Fig. 10(d). Thus, the algorithm execution time was 4 min 54 s for the proposed method, 2 h 33 min 09 s for the BSA-CM method, and 2 h 7 min 43 s for the BoB method. The proposed method achieved an algorithm execution time that was 97% faster than that of the BSA-CM method and 96% faster than that of the BoB method.

B. COMPARISON OF CLEANING TIME ON THE CHANGE TO THE STRUCTURE OF THE MAP

To analyze the cause of the drastic decrease in the cleaning time obtained for the proposed method, the shapes of the obstacles and the map sizes were changed for the three methods. In Fig. 11, there are 10 new maps used in the experiment. For each of these 10 maps, Map n1 ∼ Map n10, the map size was increased from 100×100 to $1,000 \times 1,000$. At the same time, obstacles of various shapes were utilized, and the number of obstacles was increased randomly. The above results showed that the BoB method was superior to the BSA-CM method. Therefore, the following experiments compared the BoB method and the proposed method.

Fig. 12 shows the cleaning times of the BoB method and the proposed method by applying different numbers of cleaning robots for the 10 maps from Fig. 11. From analyzing the experimental results, we found that as the map size increased, the cleaning time of the proposed method decreased in comparison with the cleaning time of the BoB method. For example, for Map n5 with a map size of 500×500 using three robots, as shown in Fig. 12(a), the cleaning time was 11 h 24 min 54 s for the proposed method and 12 h 13 min 21 s for the BoB method. Thus, the proposed method reduced the cleaning time by 34 min and 57 s as compared to the BoB method. For Map n10 with a map size of $1,000 \times 1,000$, the cleaning time of the proposed method was 50 h 27 min 12 s, as compared to, the cleaning time of the BoB method of 55 h 15 min 32 s. Consequently, the cleaning time of the proposed method was shorter by 4 h 48 min 20 s than that of the BoB method.

Moreover, for Map n10 with a map size of $1,000 \times 1,000$, using four, five, and six robots, the cleaning times of the proposed method were 35 h 39 min 21 s, 27 h 11 min, and 27 h 33 min 14 s, respectively. The cleaning times of the BoB method were 51 h 31 min 20 s, 40 h 12 min 54 s,



(a) Map n1 (100 × 100) (b) Map n2 (200 × 200)

(c) Map n3 (300 × 300) (d) Map n4 (400 × 400)

(e) Map n5 (500 × 500) (f) Map n6 (600 × 600)

(g) Map n7 (700 × 700) (h) Map n8 (800 × 800)

(i) Map n9 (900 × 900) (j) Map n10 (1, 000 × 1, 000)

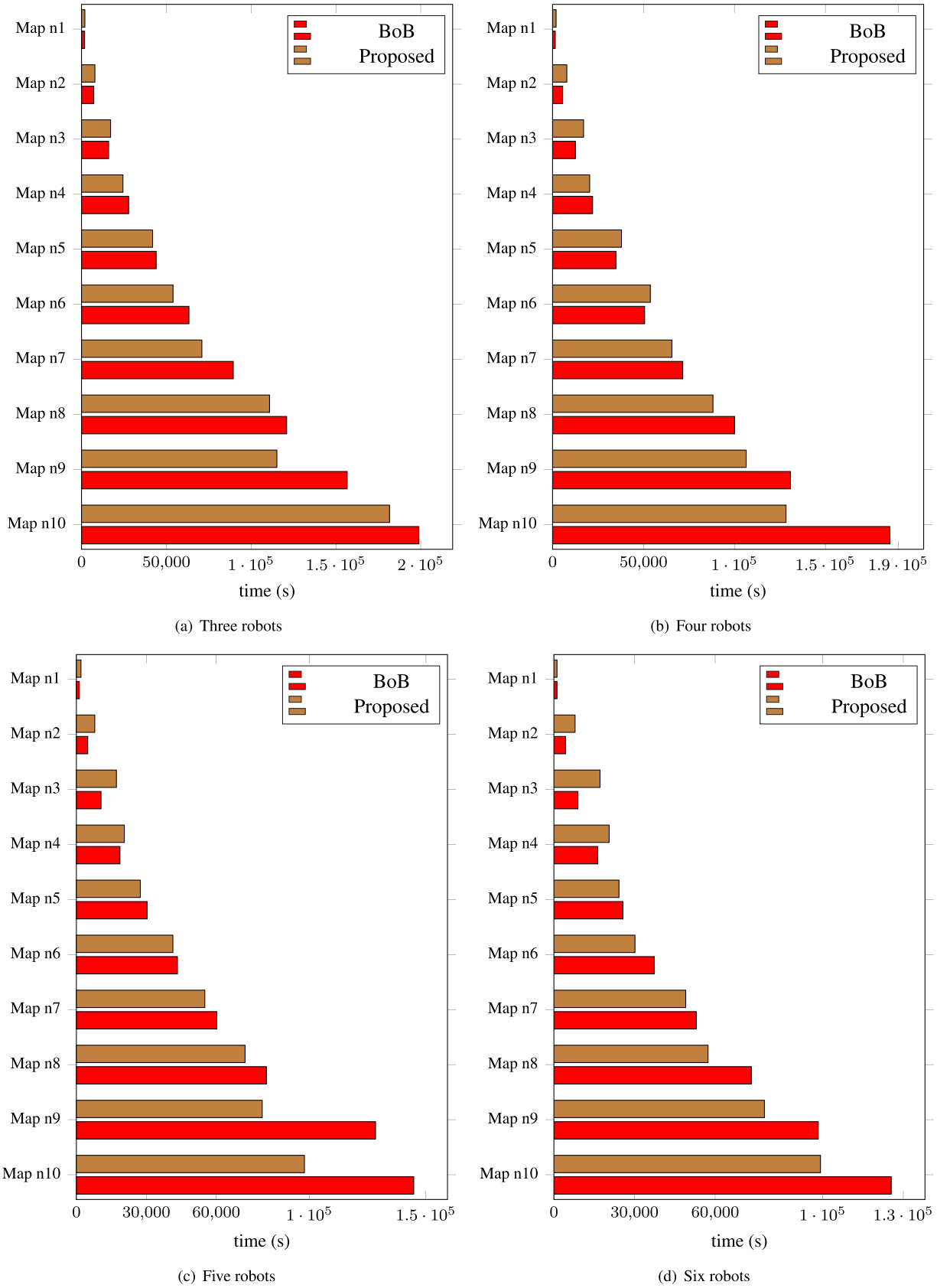
FIGURE 11. Ten maps with various shapes of obstacles while increasing the size of the map.

and 34 h 33 min 14 s, respectively. Thus, the proposed method reduced the cleaning time by 15 h 51 min 59 s, 13 h 1 min 54 s, 7h 18 min 42 s, respectively. As can be seen from the results, the decreased cleaning time in the proposed method as compared to the BoB method could be attributed to the algorithm execution times. Thus, Fig. 13 shows that the algorithm execution time obtained using the proposed method was significantly faster than the algorithm execution time obtained using the BoB method for the 10 maps with the sizes of 100 × 100 to 1, 000 × 1, 000.

The algorithm execution time measured by the proposed method was shorter than that measured by the BoB method as the proposed method efficiently solved the limitations of the BoB method. First, the proposed method reduced the amount of information-sharing by using the sub-maps, and the independence among the multi-robots increased. Thus, the idle time of the cleaning robot decreased the algorithm execution time. In the case of Map n1 with a map size of 100 × 100 in Fig. 13(a), the idle time of the robot was 166 s (2 min 46 s) in the BoB method and 1 s in the proposed method. Therefore, the idle time for the proposed method was 99% shorter than that for the BoB method. In the case of Map n5 with a map size of 500 × 500 in Fig. 13(e), the idle time for the BoB method was 6,321 s (1 h 45 min 21 s). Moreover, the idle time for the proposed method was 39 s. Thus, the idle time of the robot for the proposed method was 99% less than that for the BoB method. In Fig. 13(j), in Map n10 with a map size of 1, 000 × 1, 000, the idle time of the robot was calculated to be 62,384 s (17 h 19 min 44 s) in the BoB method. Moreover, the proposed method measured the idle time of the robot to be 454 s (7 min 34 s). Therefore, the proposed method had an idle time that was 99% shorter than that for the BoB method.

Second, the corner and the edge information of the sub-map did not increase with an increase in the map size when the cleaning robot selected the sub-map in the proposed method. Thus, the cleaning-distribution time of the proposed method was reduced as compared to that of the previous methods using the grid information. As shown in Fig. 13, in the case of Map n1 with a map size of 100 × 100, the cleaning-distribution time was 4 s for the BoB method and 1 s for the proposed method. The cleaning-distribution time was 177 s (2 min 57 s) in the BoB method for Map n5 with a map size of 500 × 500. Moreover, it took 31 s in the proposed method. For Map n10 with a map size of 1, 000 × 1, 000, the cleaning time of the BoB method and that of the proposed method were 3,633 s (1 h 33 s) and 811 s (13 min 31 s), respectively. For the 10 maps shown in the figure, on average, the proposed method had a 60% faster cleaning-distribution time than the BoB method did.

In the proposed method, the ratio of the sub-map decomposition time was analyzed. The BoB method does not use a sub-map, and therefore the sub-map decomposition time was 0 s in the 10 maps depicted in Fig. 13. Whereas, in the case of the proposed method, the sub-map-decomposition time was 0.2 s when the map size was 100 × 100. The sub-map decomposition time was 2 s when the size of the map was 500 × 500. In addition, the proposed method



(a) Three robots

(b) Four robots

(c) Five robots

(d) Six robots

FIGURE 12. Comparison of the cleaning time between the BoB method and the proposed method.

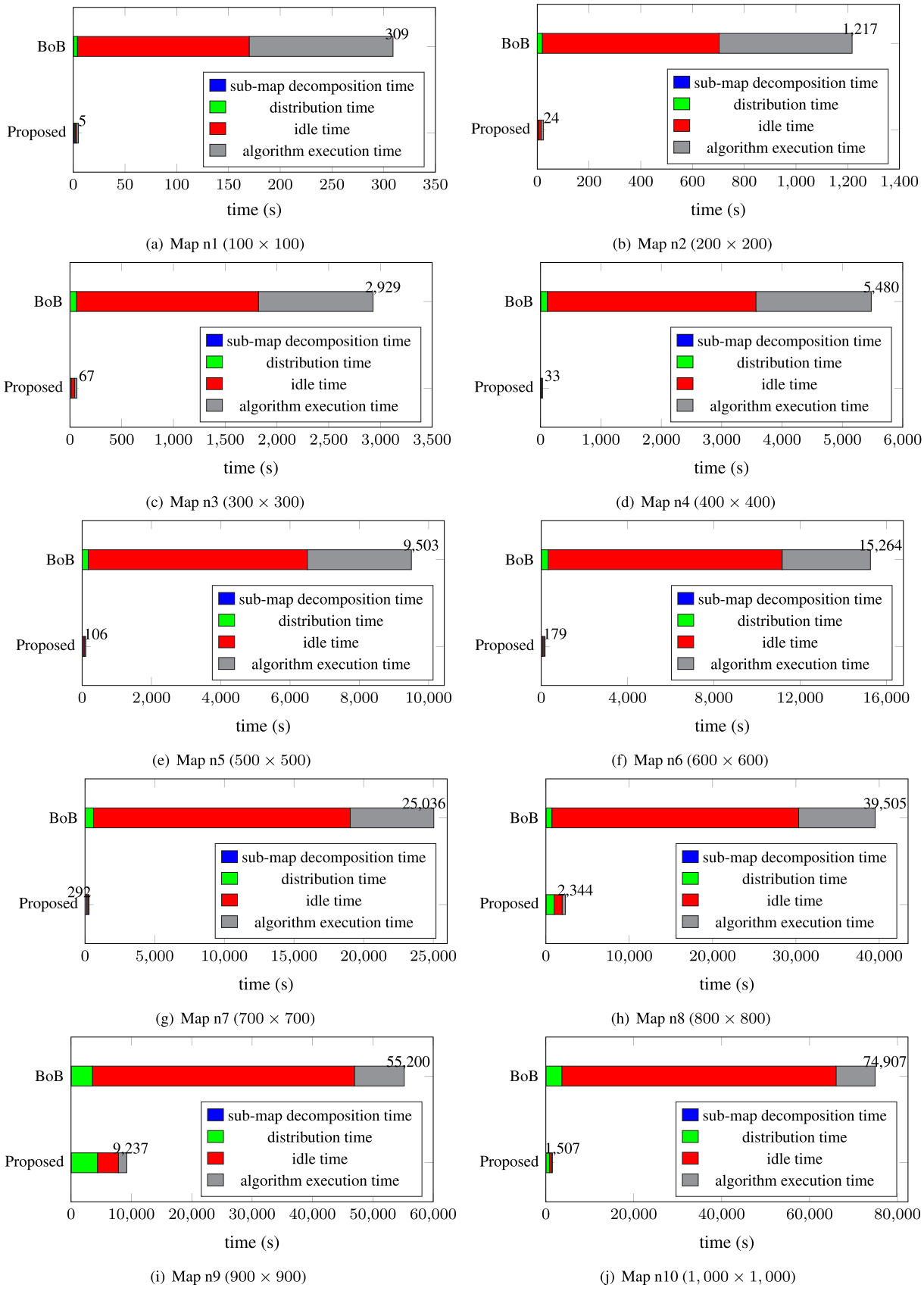
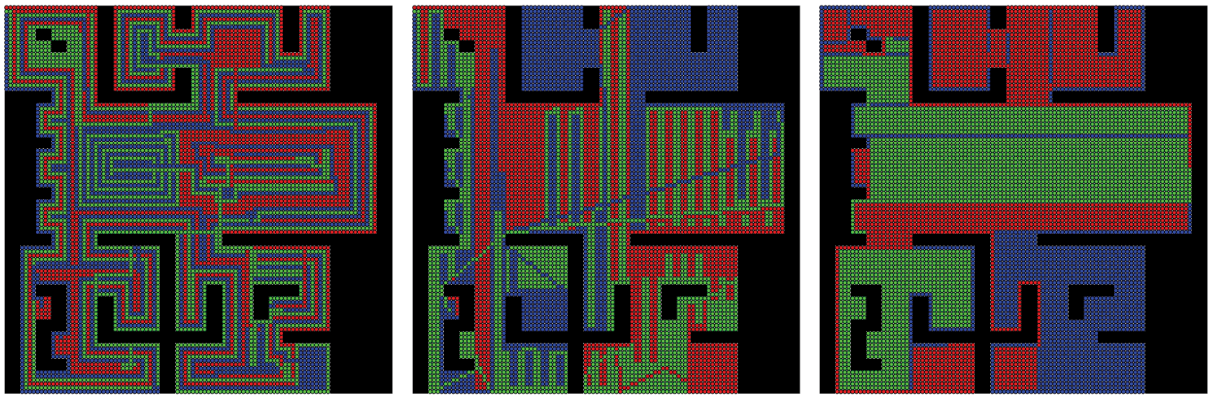


FIGURE 13. Comparison of the detailed algorithm execution time between the BoB method and the proposed method using three cleaning robots.



(a) BSA-CM method using spiral paths (b) BoB method using zigzag paths (c) Proposed method using spiral paths

FIGURE 14. Comparison of simulation results of the three methods using three cleaning robots.

had 6 s of the sub-map decomposition time in the map size of $1,000 \times 1,000$. Nevertheless, the sub-map decomposition time was a small proportion of the algorithm execution time obtained using the proposed method.

The simulation of the two previous methods and the proposed method is shown in Fig. 14. This figure shows the path of each robot when the two previous methods and the proposed method were applied to three multi-cleaning robots in map 02 with a map size of 100×100, as shown in Fig. 8(b). The paths of each robot for the three robots are indicated by red, green, and blue. The BSA-CM method used a spiral path, and the zigzag path was applied to the BoB method. The spiral paths using the potential method were employed in the proposed method. We have provided a simulation video to show the cleaning process and the path of the proposed method more precisely; it can be found at http://ibot.knu.ac.kr/vediomrcpp.html.

V. CONCLUSION

In this paper, we proposed an MRCPP method that could be applied in unknown large environments. The proposed method classified multi-cleaning robots into R^m and R^s, in which R^m decomposed the entire map into several sub-maps by using the proposed map decomposition method. Additionally, each R^s utilized a spiral coverage path planning method based on a potential graph to clean each sub-map area. Furthermore, the problem of collisions between the robots in the cleaning process was solved using the proposed cleaning distribution method. Through this method, the cleaning time was proved to be shortened than that of the previous MRCPP methods used for maps with various sizes and different shapes of obstacles. In particular, the algorithm execution time was considerably reduced in the total cleaning time. The proposed method decreased the total cleaning time due to the reduction of the algorithm execution time; however, the travel time of the robot was longer than that in the previous MRCPP methods. This fact did not affect the total cleaning time. However, if the travel time of the robot could be reduced further, the proposed method could be completed within an even shorter cleaning time. In the future, we plan to continue studying techniques for reducing the travel time and the number of turns using the proposed method. Furthermore, we plan to improve the algorithm to shorten the cleaning time further for more substantial environments. Moreover, problems such as low battery and sudden failure may occur when using a commercial cleaning robot for a large map. A multi-cleaning robot system in which the cleaning robots can assist each other when these problems occur will be added to the proposed method.

REFERENCES

[1] S.-H. Nam, I.-S. Shin, J.-J. Kim, and S.-G. Lee, "Complete coverage path planning for multi-robots employing flow networks," in *Proc. Int. Conf. Control, Autom. Syst.*, Oct. 2008, pp. 2117–2120.

[2] A. Nikitenko, J. Grundspenkis, A. Liekna, M. Ekmanis, G. Kulikovskis, and I. Andersone, "Multi-robot system for vacuum cleaning domain," in *Proc. Int. Conf. Practical Appl. Agents Multi-Agent Syst.* Cham, Switzerland: Springer, 2014, pp. 363–366.

[3] I. Shnaps and E. Rimon, "Online coverage of planar environments by a battery powered autonomous mobile robot," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 425–436, Apr. 2016.

[4] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auto. Syst.*, vol. 61, no. 12, pp. 1258–1276, Dec. 2013.

[5] N. Kamra, T. K. S. Kumar, and N. Ayanian, "Combinatorial problems in multirobot battery exchange systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 852–862, Apr. 2018.

[6] S. Opfer, H. Skubch, and K. Geihs, "Cooperative path planning for multi-robot systems in dynamic domains," in *Mobile Robots-Control Architectures, Bio-Interfacing, Navigation, Multi Robot Motion Planning and Operator Training*. Rijeka, Croatia: InTech, 2011, pp. 237–258.

[7] C. Wei, K. V. Hindriks, and C. M. Jonker, "Altruistic coordination for multi-robot cooperative pathfinding," *Appl. Intell.*, vol. 44, no. 2, pp. 269–281, Mar. 2016.

[8] A. Khan, I. Noreen, and Z. Habib, "On complete coverage path planning algorithms for non-holonomic mobile robots: Survey and challenges," *J. Inf. Sci. Eng.*, vol. 33, no. 1, pp. 101–121, 2017.

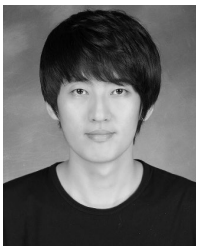
[9] E. Gonzalez and E. Gerlein, "BSA-CM: A multi-robot coverage algorithm," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol.*, Sep. 2009, pp. 383–386.

[10] H. H. Viet, V.-H. Dang, S. Choi, and T. C. Chung, "BoB: An online coverage approach for multi-robot systems," *Int. J. Speech Technol.*, vol. 42, no. 2, pp. 157–173, Mar. 2015.

[11] E. Gonzalez, M. Alarcon, P. Aristizabal, and C. Parra, "BSA: A coverage algorithm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 2, Oct. 2003, pp. 1679–1684.

[12] J. Song and S. Gupta, "ε^*: An online coverage path planning algorithm," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 526–533, Apr. 2018.

[13] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *Int. J. Adv. Robotic Syst.*, vol. 10, no. 12, p. 399, Jan. 2013.
[14] X. Miao, J. Lee, and B. Y. Kang, "Scalable coverage path planning for cleaning robots using rectangular map decomposition on large environments," *IEEE Access*, vol. 6, pp. 38200–38215, 2018.
[15] H. Choset, "Coverage for robotics—A survey of recent results," *Ann. Math. Artif. Intell.*, vol. 31, nos. 1–4, pp. 113–126, 2001.
[16] N. Hazon and G. A. Kaminka, "On redundancy, efficiency, and robustness in coverage for multiple robots," *Robot. Auto. Syst.*, vol. 56, no. 12, pp. 1102–1114, Dec. 2008.
[17] V. G. Nair and K. R. Guruprasad, "GM-VPC: An algorithm for multi-robot coverage of known spaces using generalized Voronoi partition," *Robotica*, vol. 38, pp. 1–16, Jul. 2019.
[18] H. Liu, J. Ma, and W. Huang, "Sensor-based complete coverage path planning in dynamic environment for cleaning robot," *CAAI Trans. Intell. Technol.*, vol. 3, no. 1, pp. 65–72, Mar. 2018.
[19] I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein, "Distributed covering by ant-robots using evaporating traces," *IEEE Trans. Robot. Autom.*, vol. 15, no. 5, pp. 918–933, Oct. 1999.
[20] I. A. Wagner, Y. Altshuler, V. Yanovski, and A. M. Bruckstein, "Cooperative cleaners: A study in ant robotics," *Int. J. Robot. Res.*, vol. 27, no. 1, pp. 127–151, Jan. 2008.
[21] I. Rekleitis, V. Lee-Shue, A. Peng New, and H. Choset, "Limited communication, multi-robot team based coverage," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 4, Apr. 2004, pp. 3462–3468.
[22] C. Sze Kong, N. Ai Peng, and I. Rekleitis, "Distributed coverage with multi-robot system," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2006, pp. 2423–2429.
[23] I. Rekleitis, A. P. New, E. S. Rankin, and H. Choset, "Efficient boustrophedon multi-robot coverage: An algorithmic approach," *Ann. Math. Artif. Intell.*, vol. 52, nos. 2–4, pp. 109–142, Apr. 2008.
[24] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Auton. Robots*, vol. 9, no. 3, pp. 247–253, 2000.
[25] N. Hazon, F. Mieli, and G. A. Kaminka, "Towards robust on-line multi-robot coverage," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2006, pp. 1710–1715.
[26] K. S. Senthilkumar and K. K. Bharadwaj, "Multi-robot exploration and terrain coverage in an unknown environment," *Robot. Auto. Syst.*, vol. 60, no. 1, pp. 123–132, Jan. 2012.
[27] H. Yamachi, Y. Tsujimura, and Y. Kambayashi, "Influence of field structure on the multi-agent coverage algorithm on unknown fields," *J. Adv. Comput. Intell. Intell. Informat.*, vol. 17, no. 6, pp. 883–889, Nov. 2013.
[28] H. H. Viet, V.-H. Dang, M. N. U. Laskar, and T. Chung, "BA*: An online complete coverage algorithm for cleaning robots," *Appl. Intell.*, vol. 39, no. 2, pp. 217–235, Sep. 2013.
[29] E. U. Acar and H. Choset, "Sensor-based coverage of unknown environments: Incremental construction of morse decompositions," *Int. J. Robot. Res.*, vol. 21, no. 4, pp. 345–366, Apr. 2002.
[30] D. Zhu, C. Tian, X. Jiang, and C. Luo, "Multi-AUVs cooperative complete coverage path planning based on GBNN algorithm," in *Proc. 29th Chin. Control Decis. Conf. (CCDC)*, May 2017, pp. 6761–6766.
[31] Y. Gabriely and E. Rimon, "Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, May 2002, pp. 954–960.
[32] R. Arezoumand and S. Mashohor, "Deploying clustered wireless sensor network by multi-robot system," in *Proc. IEEE Int. Conf. Control Syst., Comput. Eng. (ICCSCE)*, Nov. 2014, pp. 107–111.
[33] R. Arezoumand, S. Mashohor, and M. H. Marhaban, "Efficient terrain coverage for deploying wireless sensor nodes on multi-robot system," *Intell. Service Robot.*, vol. 9, no. 2, pp. 163–175, Apr. 2016.
[34] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," in *Cooperative Robots and Sensor Networks*. Cham, Switzerland: Springer, 2015, pp. 31–51.
[35] B. Kartal, J. Godoy, I. Karamouzas, and S. J. Guy, "Stochastic tree search with useful cycles for patrolling problems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 1289–1294.
[36] B. Kartal, E. Nunes, J. Godoy, and M. Gini, "Monte Carlo tree search with branch and bound for multi-robot task allocation," in *Proc. Workshop Auto. Mobile Service Robots (IJCAI)*, 2016, pp. 1–7.
[37] M. Garzón, J. Valente, J. J. Roldán, L. Cancar, A. Barrientos, and J. Del Cerro, "A multirobot system for distributed area coverage and signal searching in large outdoor scenarios," *J. Field Robot.*, vol. 33, no. 8, pp. 1087–1106, 2016.
[38] H. X. Pham, H. M. La, D. Feil-Seifer, and M. C. Deans, "A distributed control framework of multiple unmanned aerial vehicles for dynamic wildfire tracking," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 4, pp. 1537–1548, Apr. 2020.
[39] M. Santos, Y. Diaz-Mercado, and M. Egerstedt, "Coverage control for multirobot teams with heterogeneous sensing capabilities," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 919–925, Apr. 2018.
[40] E. Nunes and M. Gini, "Multi-robot auctions for allocation of tasks with temporal constraints," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2110–2116.
[41] B. P. Gerkey and M. J. Mataric, "Multi-robot task allocation: Analyzing the complexity and optimality of key architectures," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, Sep. 2003, pp. 3862–3868.
[42] E. Gonzalez, O. Alvarez, Y. Diaz, C. Parra, and C. Bustacara, "BSA: A complete coverage algorithm," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 2040–2044.
[43] M. Kapanoglu, M. Alikalfa, M. Ozkan, A. Yazıcı, and O. Parlaktuna, "A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time," *J. Intell. Manuf.*, vol. 23, no. 4, pp. 1035–1045, Aug. 2012.
[44] B. Sun, D. Zhu, C. Tian, and C. Luo, "Complete coverage autonomous underwater vehicles path planning based on glasius bio-inspired neural network algorithm for discrete and centralized programming," *IEEE Trans. Cognit. Develop. Syst.*, vol. 11, no. 1, pp. 73–84, Mar. 2019.
[45] A. Gautam, J. K. Murthy, G. Kumar, S. P. A. Ram, B. Jha, and S. Mohan, "Cluster, allocate, cover: An efficient approach for multi-robot coverage," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2015, pp. 197–203.
[46] Z. Zhang and Z. Zhao, "A multiple mobile robots path planning algorithm based on A-star and dijkstra algorithm," *Int. J. Smart Home*, vol. 8, no. 3, pp. 75–86, May 2014.
[47] M. J., R. L., and S. P., "Comparative analysis of search algorithms," *Int. J. Comput. Appl.*, vol. 179, no. 50, pp. 40–43, Jun. 2018.
[48] E. Gerlein and E. Gonzalez, "Multirobot cooperative model applied to coverage of unknown regions," in *Multi-Robot Systems, Trends and Development*. Rijeka, Croatia: InTech, 2011.
[49] J. Lee and D.-W. Kim, "An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph," *Inf. Sci.*, vol. 332, pp. 1–18, Mar. 2016.



XU MIAO received the Ph.D. degree in robot engineering and the M.S. degree in mechanical engineering from Kyungpook National University, Daegu, South Korea, in 2019 and 2013, respectively. His research interests include the path planning of autonomous robots, artificial intelligence for intelligent robots, and deep learning for the autonomous vehicle.

HYUN-SOON LEE received the M.S. degree in computer engineering from Kyungpook National University, Daegu, South Korea, where she is currently pursuing the Ph.D. degree with the School of Mechanical Engineering. Her research interests include the autonomous vehicle systems and artificial intelligence for intelligent robots.

BO-YEONG KANG (Member, IEEE) received the B.S., M.A., M.S., and Ph.D. degrees from Kyungpook National University (KNU), Daegu, South Korea. She was with Seoul National University as a Research Professor and held a postdoctoral position with the Korea Advanced Institute of Science and Technology (KAIST). She has been a Professor with the School of Mechanical Engineering, KNU, since 2009. Her current research interest includes artificial intelligence implementation for social and intelligent robots.