Preprint version accepted to Ad Hoc Networks, Elsevier - http://dx.doi.org/10.1016/j.adhoc.2015.12.001

Finding the Optimal Location and Allocation of Relay Robots for Building a Rapid End-to-end Wireless Communication

Byung-Cheol Min[a,*], Yongho Kim[a], Sangjun Lee[a], Jin-Woo Jung[b], Eric T. Matson[a]

[a]*Department of Computer and Information Technology, Purdue University, West Lafayette, IN, USA 47907*
[b]*Department of Computer Science and Engineering, Dongguk University, Seoul, Republic of Korea 100-715*

Abstract

This paper addresses the fundamental problem of finding an optimal location and allocation of relay robots to establish an immediate end-to-end wireless communication in an inaccessible or dangerous area. We first formulate an end-to-end communication problem in a general optimization form with constraints for the operation of robots and antenna performance. Specifically, the constraints on the propagation of radio signals and infeasible locations of robots within physical obstacles are considered in case of a dense space. In order to solve the formulated problem, we present two optimization techniques such as Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). Finally, the feasibility and effectiveness of the proposed methods are demonstrated by conducting several simulations, proof-of-concept study, and field experiments. We expect that our novel approach can be applied in a variety of rescue, disaster, and emergency scenarios where quick and long-distance communications are needed.

Keywords: end-to-end communication, communication chains, relay robots, multi robot system, evolutionary algorithm, safety, security, and rescue robotics (SSRR)

1. Introduction

In a disaster area, where previously established networks are destroyed, one of the top priorities of search-and-rescue missions is regaining or rebuilding a communication link between the base and rescuers as quick as possible in order to secure the safety of both rescuers and survivors [1]. A group of autonomous relay robots carrying wireless communication devices can be deployed to rapidly build the wireless connection between two end nodes, and thus enabling end-to-end communication[1] [2]. This would effectively give firefighters, rescuers, and first responders the ability to communicate with command center and search the best evacuation route as shown in Figure 1 [3]. Since an immediate and optimal deployment of relay robots plays a pivotal role in such an event, we tackle these deployment problems in this paper.

Given two endpoints and basic map information such as the physical location of buildings on a plane, multi robots carrying wireless devices can be deployed to relay a communication signal between two points in a cascaded communication chain. We assume that robots are initially located around one of the points, e.g., a command center, and an initial communication between the end points does not exist. With a rapid establishment of a wireless backbone is our primary goal as a robot deployment planner, the research aim is: *How do we find optimal locations and allocations of the robots to construct the end-to-end communication promptly and efficiently?*

Figure 1: A firefighter linked to command center through a set of relay robots.

For effective deployment of relay robots, we divide the deployment problem into two fundamental sub problems - *Location* and *Allocation*, which needs to be solved simultaneously. The *Location* problem consists of finding optimal locations, where networked robots need to be located to relay radio signal between two end nodes in the quickest time. The *Allocation* problem consists of finding which robots need to be assigned to each location.

*Corresponding author
Email addresses: minb@purdue.edu (Byung-Cheol Min), kim1681@purdue.edu (Yongho Kim), lee1424@purdue.edu (Sangjun Lee), jwjung@dongguk.edu (Jin-Woo Jung), ematson@purdue.edu (Eric T. Matson)

[1]Throughout this paper, the term "end-to-end network" or "end-to-end communication" refers to the communications link between two end nodes, e.g., a command center and an end (lead or exploring) node.

The problem of building communication bridge on a plane is classified as NP-hard problem [4]. The *Location* problem is approachable with continuous variables and the *Allocation* problem is approachable with discrete variables. Thus, an optimization problem can be formulated as a combinatorial problem. In this research, Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are employed to solve the optimization problem with evolutionary heuristic methods.

The primary contribution of this study is introducing a simple and effective way of applying evolutionary algorithms to robotic sensor deployment problems. Since we consider most constraints of end-to-end communication problem that can be observed in the real world, solutions found by the proposed algorithms are highly feasible and robust. In addition, as the proposed algorithms are evolution based, they are easy to add or remove robots (i.e., it is scalable), which is very important in that they can be applied to a wide range of environments. We expect that this research will play a significant role in creating a rapid and advantageous communication bridge for complex environments and will stimulate an active and vibrant research field where robotic sensor network related problems could be approached with or solved by evolutionary algorithms.

The remainder of this paper is organized as follows. First, in Section 2, we describe related works; the end-to-end communication, *Location* and *Allocation* problems, and our previous research. In Section 3, we address the basic concept of an end-to-end wireless network and formulate the fundamental problem to be solved. Also, we present additional constraints to deal with the establishment of the network in more complex environments. Then, we present two applied optimization algorithms in Section 4. Simulation results and proof-of-concept study in Section 5, and field experiments in Section 6 are shown to verify the performance of the proposed algorithm. Lastly, conclusions and future works will be summarized in Section 7.

2. Related Works

2.1. End-to-end Communication

Due to the high mobility and flexibility of operating mobile robots, those such as aerial vehicles and mobile robots have been widely used to establish or maintain ad hoc networks in field of robotics [5]. For example, a mobile unit can be used to form a desired shape of network if a wireless device is mounted on the mobile unit. Then, the mobile unit turns out to be a relay or router. Task of building end-to-end communication can be divided into two categories depend on types of node; dynamic end node and static end node.

First, building end-to-end communication for dynamic end nodes can be achieved by deploying a team of leader-follower robots in a convoying arrangement [6, 7, 8, 9, 10]. In this way, multiple robots can be used, and only the leader requires navigation capabilities to create the network while followers do not require any planning. Alternatively, they need to follow the leader or the precedent robot. Therefore, this approach is more suitable for dynamic environments where situation can be frequently changed because it is performed based on reactive approaches rather than pre-planning.

Second, building an end-to-end communication for static end nodes can be realized by planning final robot positions prior to deployment [11, 12, 13, 14, 15]. This planning should be designed to optimize the communication link, and thus this approach is suitable for a static environment rather than dynamic environments. This is also useful for cases where a rapid establishment of the network is required, because this approach does not require a search task.

Besides, as the extension to an end-to-end communication study, maximizing coverage area of mobile robot network [16], a distributed algorithm for improving coverage [17] and an algorithm for coverage [18, 19] have been studied. While these studies focus on an establishment of the optimal network, we mainly focus on building an end-to-end communication as quick as possible because this research considers that sending a group of robots out an emergency situation where recovering or rebuilding network connection has to be a top priority.

2.2. Location and Allocation Problems

In this paper, we define a problem of finding optimal positions of robots as a series of *Location* and *Allocation* problems. In order to solve *Location* and *Allocation* problems, many of researches in various areas such as industrial engineering for operation research [20, 21, 22, 23, 24] have been done. However, our approach can be novel because we consider this problems as a combination of the robot and sensor network deployment.

The problem of *Location* and *Allocation* is also known as the *multi-weber* problem or the *p-median* problem. For example, [21] tackles finding optimal locations of facilities and allocation of customers to the facilities so that the total distance customers moved and the operating expense is minimized. With consideration of obstacles and some forbidden areas, they employ a Genetic Algorithm (GA) to effectively approach this combinatorial problem. In addition to GA, a variety of approaches have been introduced to solve *Location* and *Allocation* problem; Simulated Annealing [22], Fuzzy algorithm [23], and Particle Swarming Optimization (PSO) [24]. From the point of view of the performance evaluation, the overall performance of optimization algorithm for robots is generally determined based on energy consumption, computation time, and complexity. An extensive research has been conducted to develop optimal algorithms for the performance. For instance, a number of studies have been conducted to minimize power consumption [25] or to maximize path lifetime [26]. Similarly, the researchers have focused on finding optimal placements for mobile relays [27, 28, 29]. The mobility of robots is also an important characteristic for the overall performance so that several studies also has been done [30, 31].

Nonetheless, evolutionary based algorithms such as the GA and PSO have a number of desirable properties when it comes to solving a combinatorial problem. More specifically, they are simple and fast and it is capable of finding the global minimum in general. Because the problem we tackle in this paper does not require the online operation demands (albeit a fast processing may be desirable), it is acceptable to set up the problem, run an optimization algorithm, and then implement the solution. It is for this reason that we have decided to use the evolutionary based algorithms for this robotic network deployment problem.

2.3. Our Previous Works

A great deal of basic research in the individual areas of wireless networks, localization, autonomous robotics, and self-organizing systems has been completed, along with a working prototype system [32, 33, 7]. Our research concept began by using small robotic units, and successfully demonstrated the ability to have robots perform specified actions based on the radio frequency (RF) signal received [32]. From this basic concept, we derived a more robust system of outdoor robots that were equipped with mesh access points (APs) with the goal of autonomously establishing a linear wireless broadband mesh network [33, 7]. Our previous research has shown that the linear expansion concept could stretch a network's coverage pattern. However, it is still hard to apply to complex environments where curved or high-order formations of robots are needed. Thus, research in this paper serves to transform the linear formation technology into the adaptive and flexible formation according to problem domains requiring abnormal mobile communications such as difficult, complex, or dangerous conditions.

3. Location and Allocation Problem

3.1. Problem Statement

Let L_c and L_e be two fixed locations given on an open (x, y) coordinate plane. L_c is the location of the command center node that represents the source of the wireless signal. L_e is the location of the end node that represents the destination of the wireless signal, as shown in Figure 2. There are also n variable relay locations (L_1, L_2, ..., L_n) that are to be determined and n robots with initial known locations (R_1, R_2, ..., R_n) to be allocated to the variable relay locations. For simplicity, let n refer to the number of robots as "$Robot_1$", "$Robot_2$", "$Robot_3$", ..., "$Robot_n$".

The number of locations and robots is then n, and the number of possible allocation cases becomes $n!$. Since the number of possible cases increases exponentially as n increases, it is almost impossible to mathematically obtain explicit solutions to the *Allocation* problem. As a matter of fact, [14, 34] proposed an approximation algorithm to find the solution to the *Location* and *Allocation* problem theoretically, but they assumed the algorithm would only be applied in an open space, without any obstacles. Therefore, if there are obstacles introduced, the algorithm does not work. Instead of using approximation algorithms, we approach such problems with heuristic search methods.

Given the defined parameters above, we can set the goal of this problem to be finding:

1) The locations of relay nodes (L_1, L_2, ..., L_n) that connect L_c to L_e
2) The allocation of robots to those locations,

such that the total distance between the robots' initial locations and the variable locations is minimized. The distance, for example, can be calculated with the sum of dotted lines depicted in Figure 2 (b). Given the initial locations of $\overline{R}_i$ robot, this problem is then formulated as follows:

$$\text{Minimize}: \; f(\mathbf{x}) = \sum_{i=1}^{n} \text{dist}(\overline{R}_i, L_i) \tag{1}$$

$$\begin{aligned} \text{Subject to:} \quad & \text{dist}(L_c, L_1) \le O_r \\ & \text{dist}(L_i, L_{i+1}) \le O_r \qquad i \in \{1, 2, \ldots, n-1\}, \\ & \text{dist}(L_n, L_e) \le O_r \end{aligned} \tag{2}$$

$$\sum_{i=1}^{n} z_{ij} = 1 \qquad j \in \{1, 2, \ldots, n\}, \tag{3}$$

$$\begin{aligned} & L_i = \{(x, y) | x_{\min} \le x \le x_{\max},\; y_{\min} \le y \le y_{\max}\} \\ & i \in \{1, 2, \ldots, n\} \end{aligned} \tag{4}$$

where,

$$\mathbf{x} = \begin{bmatrix} \text{index of } Robot_i \text{ allocated to } L_i \\ \vdots \\ \text{index of } Robot_n \text{ allocated to } L_n \\ x \text{ coordinate of } L_1 \\ y \text{ coordinate of } L_1 \\ \vdots \\ x \text{ coordinate of } L_n \\ y \text{ coordinate of } L_n \end{bmatrix},$$

$i \in \{1, 2, \ldots, n\}$,

dist() = the Euclidean distance on Cartesian coordinate,

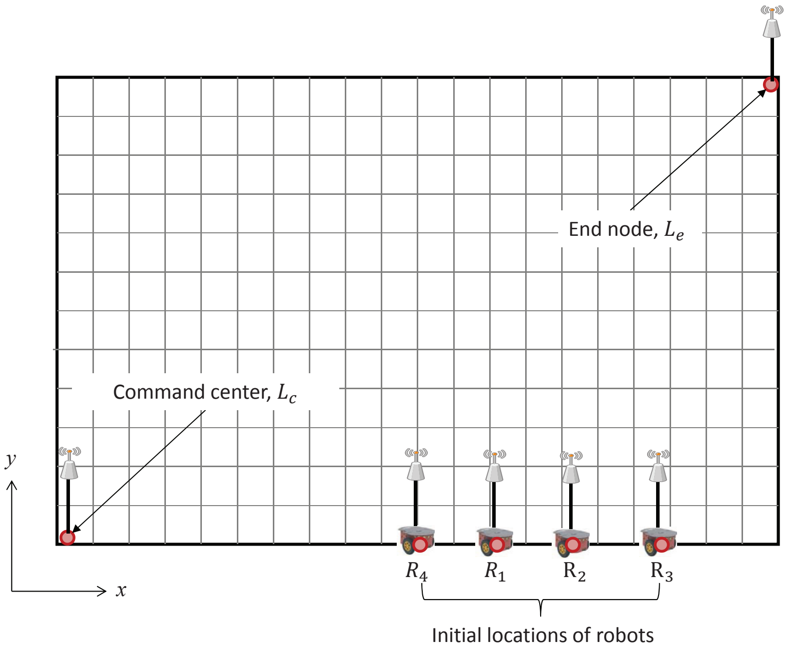
L_i = location of relay node to be determined,

$\overline{R}_i$ = initial location of the robot that will be allocated to L_i, (e.g., if $Robot_4$ is allocated to the first node, then R_4 becomes $\overline{R}_1$, as expressed in Figure 2. It can be obtained by "If-Else" statement with Algorithm 1.),

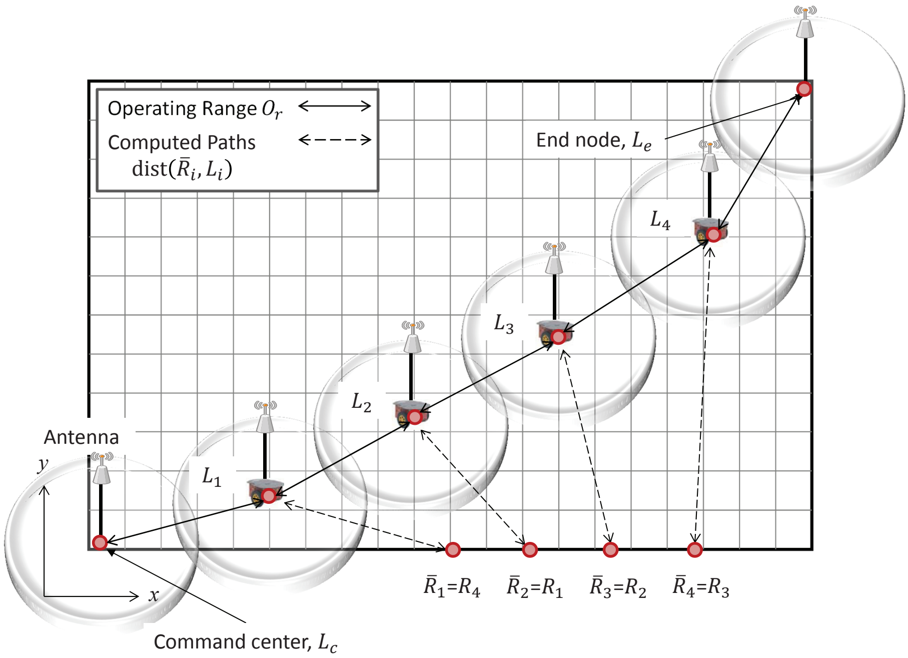
$$z_{ij} = \begin{cases} 1, \text{ the } i\text{-th robot } (Robot_i) \text{ is allocated to the } j\text{-th node } (L_j) \\ 0, \text{ otherwise,} \end{cases}$$

$x_{min}, x_{max}, y_{min}, y_{max}$ represent the size of a map.

It is worth noting that minimizing the total distance may not guarantee that the networked mobile robots carrying wireless devices build the end-to-end communication in the quickest time, because the time required to complete building the communication link may depend on the robot whose the initial location is the farthest away from its allocated (variable) location. Nonetheless, since all robots will always move to their nearest locations as possible, it is quick enough to meet the goal we set up. Furthermore, it will enable robots that arrive at their locations can immediately start doing a secondary task, for example, gathering surrounding information (e.g., hazmat) with sensor units as a surveillance or exploration robot, while waiting the last robot reaches its allocated location. Once all robots



(a) Given problems



(b) After solving problems

Figure 2: Example scenario: networked robots are deployed to connect two end nodes, creating end-to-end communication with the minimum traveling distance: (a) Four mobile robots and two ends points (a command center node and an end node) are given in a known environment; (b) Locations and allocations of the robots are determined by an algorithm proposed in this paper.

arrive at their allocated locations, then they start doing their primary task, i.e., building the end-to-end communication link.

In order to realize the goal of this problem in a real world situation, we have made following assumptions: 1) Every relay robots are initially connected, 2) One of the relay robots should be initially located within the area where a wireless signal is reachable from the command center, and 3) The command center has a processing capability. With the assumptions 1) to 3), the optimal problem is calculated on the command center with a centralized concept offline, the calculated destinations are respectively sent from the command center to each relay robot, and then every robot starts moving to their destination. This

Algorithm 1 "If-Else" statement for robot allocation problem. Note that % marks a beginning of a comment.

```
for i = 1 : n do
    if x(i) == 1 % if Robot_1 is allocated to i-th node then
        R̄_i = R_1
    else if x(i) == 2 % if Robot_2 is allocated to i-th node then
        R̄_i = R_2
    else if x(i) == n − 1 % if Robot_{n−1} is allocated to i-th node then
        R̄_i = R_{n−1}
    else if x(i) == n % if Robot_n is allocated to i-th node then
        R̄_i = R_n
    end if
end for
```

approach would bring a benefit to saving energy of the relays robots, and to extending the network lifetime because the entire optimization is calculated on the command center, and the relay robots do not consume any of energy due to the optimization process.

In addition to the three assumptions, we have an additional assumption that the map where robots will work to establish an end-to-end communication link is pre-known. Actually in most of the disaster situations, maps can be provided in advance, although there is a possibility that some parts of the map could be altered due to disasters. However, even if there are some alternations in the map, this cannot be found before the completion of actual exploration. Because of that, it would be the best to approach the *Location* and *Allocation* problem based on the pre-known maps. Therefore, we approach the problem with the pre-known maps and then if mobile robots observe any obstacles or some altered environments while moving towards their destination, each robot could run their obstacle avoidance algorithm, alter their pre-planned paths, and finally reach their destination.

Since the main goal of this research is to relay radio signals with antennas that are affixed to networked robots, we must take the constraints of the antenna performance, such as operating range, into account. So, this problem contains inequality constraints that restrict the maximum distance between two adjacent nodes, as stated in Eq. (2). In Eq. (2), O_r is the maximum (or allowable) operating range between two neighboring robots and should be set to be larger than

$$O_r \gg \left[\frac{\mathrm{dist}(L_c, L_e)}{n+1}\right]. \tag{5}$$

For instance, L_c and L_e are set to (0.0, 0.0) and (100.0, 100.0) in an open space, and five links are established by four relays (one link is made of two relay nodes, as seen in Figure 2 (b)), then O_r has to be greater than 28.284.

Equality constraints in Eq. (3) state that every robot should be allocated to a different node. Eq. (4) states that every relay node should be bounded in a map (i.e., in the operating environment).

Algorithm 1 shows an "If-Else" statement for the robot allocation problem, when there are more than three robots employed ($n \geq 3$). Note that if two robots are employed, there are only two possible solutions to the *Allocation* problem (i.e., case 1. $\overline{R}_1 = R_1$ and $\overline{R}_2 = R_2$, or case 2. $\overline{R}_1 = R_2$ and $\overline{R}_2 = R_1$). In this case, there is only one "**else if**" statements included in Algorithm 1.

In Eq. (1), $\mathbf{x}$ is a vector that includes decision variables for the optimization problem. Two examples of $\mathbf{x}$ found by Eq. (1), satisfying all constraints Eqs. (2) – (4), are presented in Eq. (6) and Eq. (7), and their final deployments are depicted in Figure 3, with full notations. In the figures, minimizing the sum of the distance of the dotted red lines is the objective of the optimization.

$$\mathbf{x} = \left[\overbrace{2\ 1\ 3}^{Allocation}\ \overbrace{59.72\ 24.00\ 124.31\ 70.68\ 182.56\ 124.31}^{Location}\right], \text{when } n = 3 \tag{6}$$

$$\mathbf{x} = \left[\overbrace{1\ 3\ 2\ 4}^{Allocation}\ \overbrace{55.24\ 23.17\ 125.10\ 40.78\ 162.21\ 82.15\ 199.50\ 136.38}^{Location}\right], \text{when } n = 4 \tag{7}$$

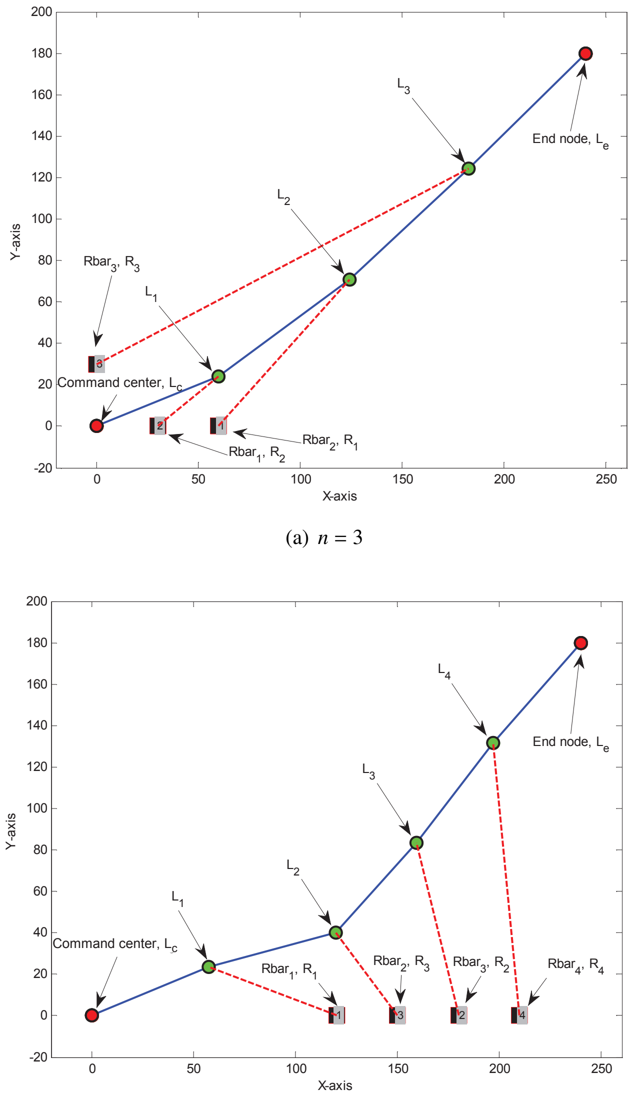
As shown in the examples above, the first three and four elements represent *Allocation* with discrete variables, indicating the index of robots, and the rest of the elements represent *Location* with continuous variables (in Cartesian coordinates). Because of these mixed variables, this problem is classified as a combinatorial problem, as mentioned earlier.

3.2. Additional Constraints

3.2.1. Dense Space

The *Location* and *Allocation* model given in the previous subsection considers the ideal case, where there are no physical obstacles. It is feasible that end-to-end communication may need to be established in an open space. In practice, however, obstacles or forbidden regions must be taken into consideration. For example, buildings, trees, and cars can be regarded as physical obstacles, in this research. Thus, we introduce additional possible constraints in this section.

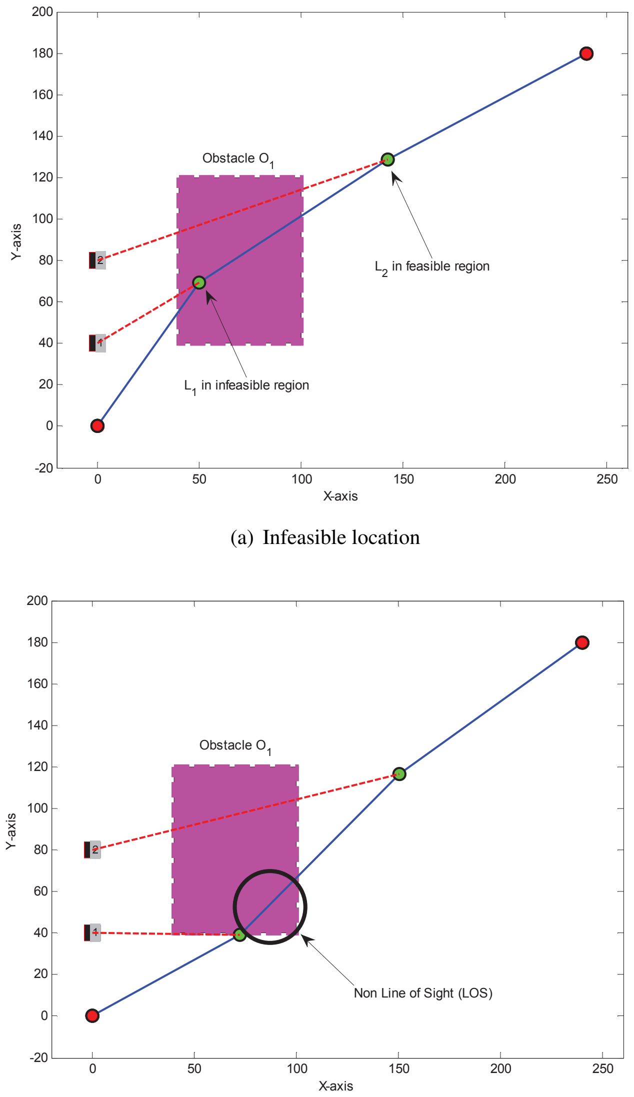
First, it is apparent that the location of a relay node cannot be within the region of obstacles, as robots cannot reside in that region. This situation is depicted in Figure 4 (a). Second, considering the propagation of radio signals from antennas, it would be much better if line-of-sight is guaranteed between adjacent nodes. If this constraint is not taken into consideration, the radio signal may be blocked by the physical obstacles (we call



(a) $n = 3$

(b) $n = 4$

Figure 3: Two examples of mobile robot deployment: (a) When there are three networked robots; (b) When there are four networked robots.



(a) Infeasible location

(b) Non-line-of-sight

Figure 4: Two cases of improper establishment of network: (a) Robots cannot be located within the region of obstacles; (b) Non-line-of-sight deteriorates propagation of radio signal from antennas.

this case as "non-line-of-sight"), as depicted in Figure 4 (b), and this may result in substantial energy loss. Furthermore, if directional antennas are used for transmissions like in [35, 36], guaranteeing line-of-sight becomes more essential. More detailed information on this line-of-sight consideration can be found in [37, 35, 38].

For the first constraint, let O_k be an obstacle region (k = 1, 2, . . . , m), where m is the number of obstacles. Then, we can define the location constraint as follows,

$$L_i \notin \bigcup_{k=1}^{m} O_k, \quad i \in \{1, 2, \ldots, n\}, \tag{8}$$

where n is the number of robots. Eq. (8) states that the locations of any relay node should not be within any obstacles.

For the second constraint, let Os_k^u be a line segment of the obstacle (u = 1, 2, . . . , l and k = 1, 2, . . . , m), where l is the number of line segments (vertices) of the k-th obstacle, and m is the number of obstacles. In addition, let Rs^v be a line segment of the two adjacent relay nodes (v = 1, 2, . . . , n+1). For example, Rs^1 is that of (L_c, L_1), Rs^2 is that of (L_1, L_2), and so on. Then, we can define the line-of-sight constraint as follows,

$$Os_k^u \cap Rs^v = \phi. \tag{9}$$

Eq. (9) denotes that there is no intersection between the line segments of obstacles and the line segments of the two adjacent relay nodes.

If Eq. (9) is imposed as a constraint in the optimization problem, the location constraint in Eq. (8) is not needed, because Eq. (9) will not tolerate any intersections between the line segments of obstacles and the line segments of the two adjacent relay nodes. If one relay node is located within the region of an obstacle, as expressed in Figure 4 (a), at least one intersection takes place and will be eliminated by Eq. (9). Yet, there would be some cases where there would be no need to impose the line-of-sight constraint, leaving only the obstacle constraint (e.g., when obstacles barely affect the radio propagation property of antennas, because they are composed of low-density material, such as wood or glass). Therefore, Eq. (8) is necessary for this research.

3.2.2. Intersection Elimination

To fulfill the robot's main role of carrying a relay, the networked robots will depart from $\overline{R}_i$ and eventually be located at L_i, by tracking the computed paths between $\overline{R}_i$ and L_i. In many cases, the computed paths may form intersections, as shown in Figure 5 (a). If paths intersect, it does not necessarily mean that a collision will take place. The robots will only collide if they reach the intersection point at the same time, but it would be better if we could make a robot tracking system simpler, by means of removing intersection points in the path planning stage. Thus, depending on the details of a given problem, it is logical to consider adding an additional constraint that guarantees there are no intersections between the computed paths. An example of successful, non-intersecting paths is depicted in Figure 5 (b).

For this intersection constraint, let $P(\overline{R}_i, L_i)$ be a set on points on the line segment connecting between i-th relay node location L_i, and its corresponding robot initial location, $\overline{R}_i$. Then, we can define the constraint as follows,

$$P(\overline{R}_i, L_i) \cap P(\overline{R}_j, L_j) = \phi, \tag{10}$$

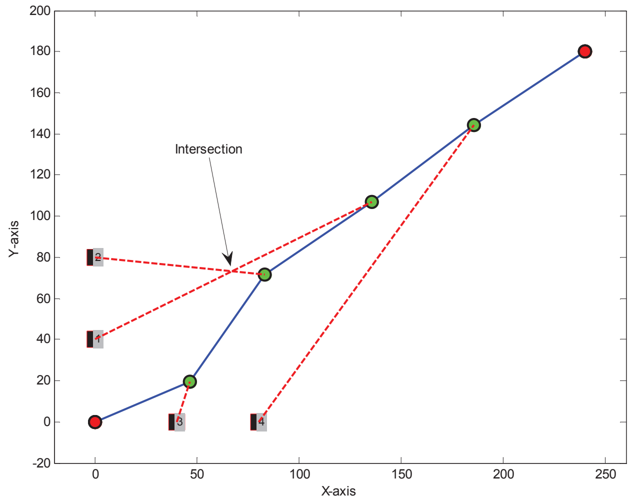
where $i \neq j$ and $i, j \in \{1, 2, ..., n\}$. Thus, Eq. (10) denotes that there are no intersections between the line segments of computed paths.

3.2.3. Shortest Path

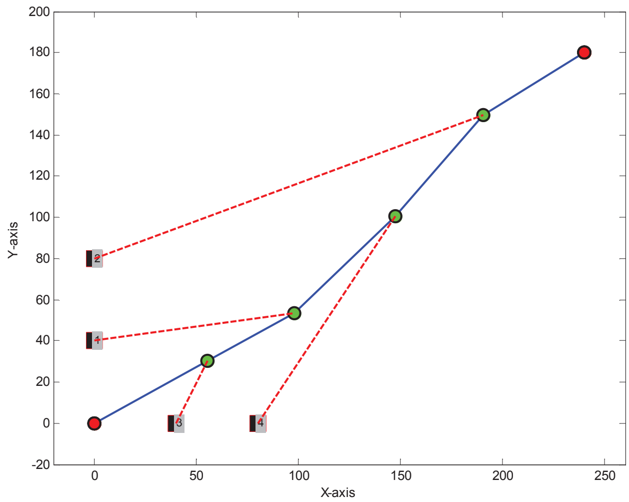
Given two points $\overline{R}_i$ and L_i at the coordinates representing an initial location of the robot and a location of a relay node, the objective function in Eq. (1) is obtained by calculating the distance of the direct line segment between the two points. However, when there are physical obstacles between the two points, the direct line segment becomes an unrealistic path for the robot, and could not be used for the objective function. In this case, shortest path algorithms, such as Dijkstra's algorithm [39] and A* algorithm [40] could be employed to obtain the realistic shortest path, while avoiding obstacles for the objective function as shown in Fig. 6.

4. Optimization Methodology

In Section 1, we stated that this robot deployment problem is a combinatorial problem that includes discrete and continuous design variables. Hence, it is appropriate to approach the problem with evolutionary heuristic methods, such as Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). In this section, we present these two optimization methods and describe how to exploit to this problem with a complete simulation.



(a) One intersection



(b) No intersection

Figure 5: Two possible solutions in the same environment: (a) There is one intersection between the computed paths for $Robot_1$ and $Robot_2$; (b) There is no intersection. Eliminating intersections can make a robot tracking system simpler.

4.1. Genetic Algorithm (GA)

GA begins by defining a chromosome to be optimized. If a chromosome c_h^t has N_{var} variables given by $x_1, x_2, \ldots, x_{N_{var}}$, then the chromosome can be described as

$$c_h^t = [x_1, x_2, \ldots, x_{N_{var}}], \ \forall h = 1, 2, \ldots, N_{pop} \tag{11}$$

where N_{pop} is a population size, and t indicates an index of iterations. In this research, the chromosome represents a decision variable vector $\mathbf{x}$, shown in Eq. (1). Thus, N_{var} can be obtained by $N_{var} = 3n$.

The chromosome is encoded to have a binary string with a different number of bits in each variable (gene). For the *Allocation* problem, the number of bits should be larger than or equal $\log_2(n)$, and for the *Location* problem, the number of bits should be carefully determined by considering a desired resolution of x-y coordinate of L_i. There is an observable relationship among the resolution, the number of bits, and bounds on variables, as follows,

$$\mathbf{r} = \frac{\mathbf{x}^U - \mathbf{x}^L}{(2^{\mathbf{b}} - 1)} \tag{12}$$

where

$\mathbf{x}^U$ = upper bound on variable, $\mathbf{x}^U = [x_1{}^U, x_2{}^U, \ldots, x^U_{N_{var}}]$

$\mathbf{x}^L$ = lower bound on variable, $\mathbf{x}^L = [x_1{}^L, x_2{}^L, \ldots, x^L_{N_{var}}]$

$\mathbf{b}$ = number of bits to code x_i ($i = 1, 2, \ldots, N_{var}$), $\mathbf{b} = [b_1, b_2, \ldots, b_{N_{var}}]$

$\mathbf{r}$ = resolution between discretized values of x_i, $\mathbf{r} = [r_1, r_2, \ldots, r_{N_{var}}]$.

An example of the encoded chromosome (when $n = 4$) is shown as follows,

$$C = [\overbrace{\underbrace{01}_{gene1} \cdots \underbrace{11}_{gene4}}^{Allocation}\ \overbrace{\underbrace{000010111100}_{gene5} \cdots \underbrace{111100101111}_{gene12}}^{Location}]. \tag{13}$$

Then, the GA is run with the following steps:

1) Define the GA parameters (e.g., population size N_{pop}, crossover probability P_c, mutation probability P_m, and termination parameters N_{qa}).

2) Randomly generate an initial population.

3) Evaluate fitness of each individual, using Eq. (1).

4) Select two parents.

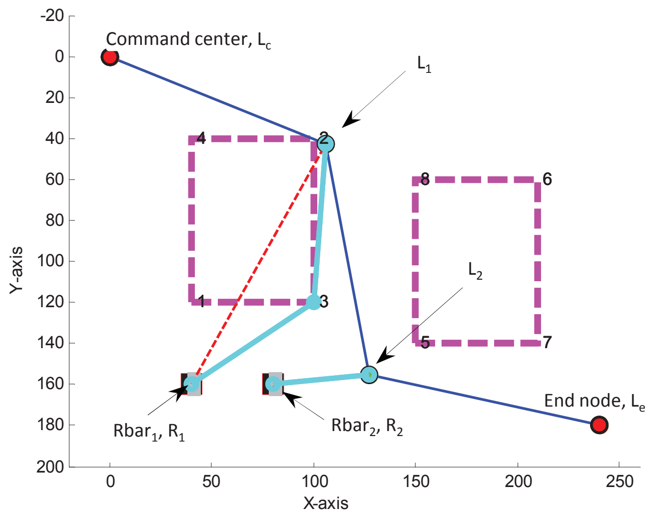
5) Crossover for two offspring.

6) Repeat step 4) and 5) until population filled.

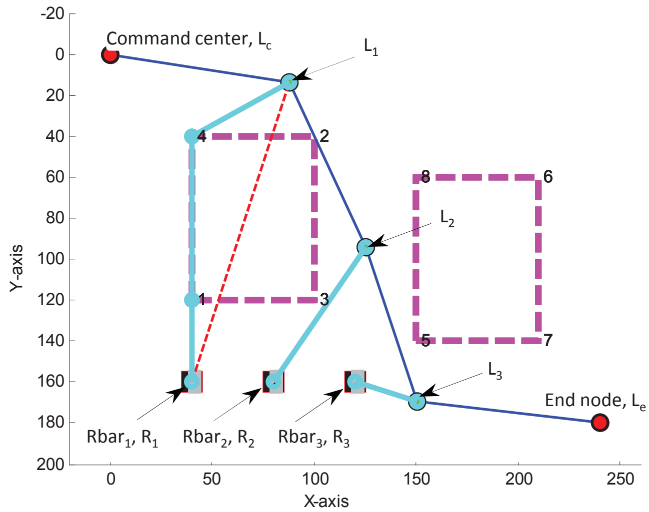
7) Examine new population for mutation.

8) Return to step 3) for next generation, until the process convergence is achieved.

In this study, population size N_{pop} is determined by $N_{pop} = 4s$, where s is a string length that can be obtained by $s = \sum_{i=1}^{n} b_i$. Our implementation of the GA uses a tournament selection method for step 4), that randomly picks a small subset of chromosomes from the mating pool, and the chromosome with the lowest cost, in this subset, becomes a parent. For step 5), we use the uniform crossover, where the first child receives a bit from the first parent with crossover probability P_c, and the second child receives a bit from the second parent. When using the uniform crossover, P_c is generally set to 0.5, like a "coin flip",



(a) Example 1



(b) Example 2

Figure 6: Examples of results achieved by Dijkstra's algorithm: sky-blue lines indicate shortest paths connecting an initial location to destination; red dashed lines indicates a direct line connecting an initial location to destination without this shortest path consideration.

generating $p = rand[0,1]$. For step 7), we randomly switch zeros and ones with mutation probability P_m. In this study, P_m is determined by $P_m = (s+1)/(2N_{pop}s)$. For step 8), the stopping criterion is activated when the total number of iterations reaches a fixed number N_{qa}.

In order to evaluate the feasibility of the GA for the robot deployment problem in this study, we implemented a simple simulation, which is described in this section. For simplicity, let us assume that there are no obstacles (i.e., it is an open space that does not require taking Eq. (8) and Eq. (9) into consideration, but intersection constraint Eq. (10) is added). Also, let n be 4, so there are four networked robots carrying wireless devices, as summarized in Table 1. Additionally, necessary settings for this first simulation are summarized in Table 2. Figure 7 (a) illustrates the given problem.

Table 1: Robot's name and initial location (when $n = 4$)

Robot's name	Initial location of robot (x,y)
$Robot_1$	$R_1 = (0.0, 30.0)$
$Robot_2$	$R_2 = (0.0, 60.0)$
$Robot_3$	$R_3 = (0.0, 90.0)$
$Robot_4$	$R_4 = (0.0, 120.0)$

Table 2: Fixed settings for the first simulation

$L_c = (0.0, 0.0)$, $L_e = (240.0, 180.0)$,
$\mathrm{x_{min}} = 0.0$, $\mathrm{x_{max}} = 240.0$, $\mathrm{y_{min}} = 0.0$, $\mathrm{y_{max}} = 180.0$

Given the settings in Tables 1 and 2, four design variables x_i, where $i = 1, \ldots, 4$, are needed to represent the robots' allocation to the nodes. For the four design variables, b_i is set to 2, $x_i^L = 1$, $x_i^U = 4$, and the resultant r_i becomes 0.53. However, we rounded the resolution $r_i = 0.53$ to $r_i = 1$ to make this variable an integer. For the *Location* problem, eight design variables x_i, where $i = 5, \ldots, 12$ are needed to represent the locations of the nodes L_i. For the four design variables' x coordinates of L_i, the coding accommodates a minimum value of 0.0 (this value indicates x_{min} in Eq. (4)), a maximum value of 240.0 (this value indicates from x_{max}), and a resolution of 0.058, using 12 bits. Similarly, for the four design variables' y coordinates of L_i, the coding accommodates a minimum value of 0.0 (this value indicates y_{min}), a maximum value of 180.0 (this value indicates y_{max}), and a resolution of 0.044, using 12 bits. It is worth noting that the number of bits can be decreased or increased. If it is decreased, the resolution of the solution will become coarser, but the process will take less time Conversely, if the number is increased, the resolution of the solution will become finer, but processing time will be increased. Using these parameters, the prescribed population size N_{pop} is then set to 416, and the prescribed mutation rate P_m is set to 0.0012. The maximum number of iterations N_{qa} is set to 150.

There are two different scenarios carefully considered in this simulation. The first scenario is when O_r is set to 60.2, which is slightly longer than the exact operating range calculated with the right side of Eq. (5). As a matter of fact, the exact range calculated in this example is 60.0, but we add 0.2 to it because GA does not find the exact solution, in general, but does find approximate solutions. Given this pre-setting, if solutions to the *Location* problem make all nodes L_i lie on the straight line that connects L_c to L_e, it can be said that this result confirms the solutions satisfy the operating range constraint in Eq. (2) very efficiently, as well as minimizing the sum of the distances between the robots and the final nodes.

The second scenario is when O_r is set to 70.0, which is large enough to satisfy a condition mentioned in Eq. (5). Given this pre-setting, if solutions to the *Location* problem make all nodes L_i closer to the robots $\overline{R}_i$, while satisfying the operating range constraint, it can be said that this result confirms that the solutions effectively minimize the sum of the distances between the robots and the final nodes.

The results of this simulation are depicted in Figure 7. More specifically, figures 7 (b) to (f) show the movements of genes marked with "o" at the different iterations of the first scenario. As shown in the figures, every gene is well-bounded, which satisfies the size constraint in Eq. (4), and every gene converges into solutions as time progresses. Until the 10th iteration, the *Allocation* problem was not solved, as $Robot_1$ was not allocated to any nodes, and $Robot_3$ was allocated to two nodes. Also, until the 30th iteration, there were intersections between $\overline{R}_i$ and L_i, which violated the intersection constraint described in Eq. (10). However, at the 90th iteration, those two constraints are no longer violated. After the 90th iteration, the solutions started trying to minimize the distances between $\overline{R}_i$ and L_i, as expressed in Eq. (1). From these patterns of convergence, we could ascertain that the GA initially tried to solve the *Allocation* problem by getting out of the domain where *Allocation* problem-related constraints are violated. After that, the GA focuses on solving the *Location* problem to minimize the distances, while satisfying all constraints.

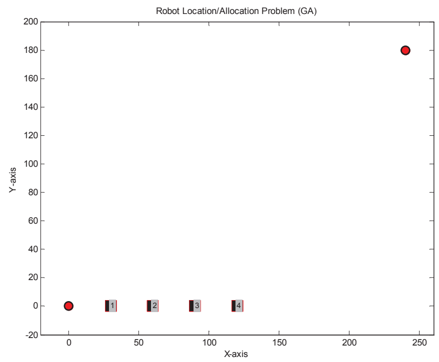
The final results of the first scenario, using the GA, are shown in Figure 7 (g). This figure was captured at the 150th iteration (recall that we set $N_{qa} = 150$). In this figure, minimizing the sum of the distance of the four dotted red lines is the objective, and the value of the sum is shown in the top left corner. One can notice that there are three "Success" messages at the top. These indicate whether the solutions can satisfy constraints or not. Therefore, there could be "Failure" messages at the top as well, which would indicate the solutions violate some constraints. The first message relates to the operating constraint in Eq. (2), the second message relates to the line-of-sight constraint in Eq. (9), and the last message relates to the intersection constraint in Eq. (10). Also, numbers on the bottom show the distances of each link that connects adjacent nodes from L_c to L_e. From these numbers, we can determine which links violate the operating range constraint.

From Figure 7 (g), it can be seen that the determined nodes L_i form an almost straight line connecting L_c to L_e, as we expected, and every robot is allocated to different nodes L_i, while satisfying all constraints that we imposed. Figure 7 (h) shows the result of the second scenario. From this figure, it can be seen that the final nodes L_i are closer to the robots $\overline{R}_i$ than in the first scenario, while satisfying all constraints, as well as minimizing the sum of the distances.

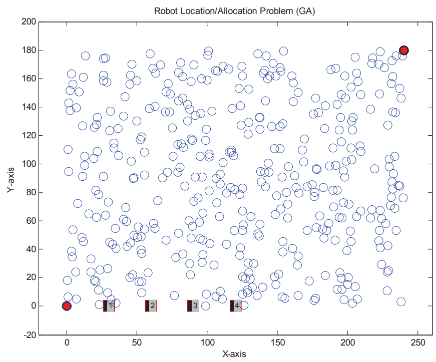
From these two simulations, it has been shown that the GA is capable of solving the problem of the robot deployment in an open space. However, as we have dealt with relatively easy problems that have no obstacles and no additional constraints, it is not possible to conclusively say that our GA can fully solve the optimization problem that we formulated in this study. Therefore, we will examine the GA with more complex problems in Section 5.

4.2. Particle Swarm Optimization (PSO)

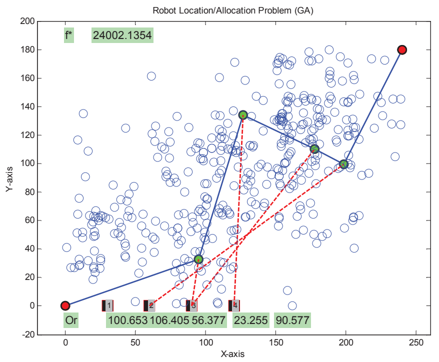
PSO is based on swarm behavior observed in nature, such as fish flocking and birds schooling. PSO shares some important attributes with GA, in that these two evolutionary algorithms are population-based search methods. In other words, PSO and GA move from a set of population to another set of population



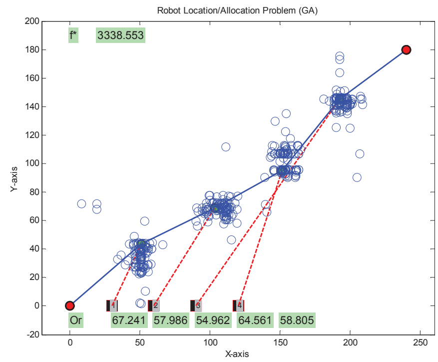
(a) Given problems



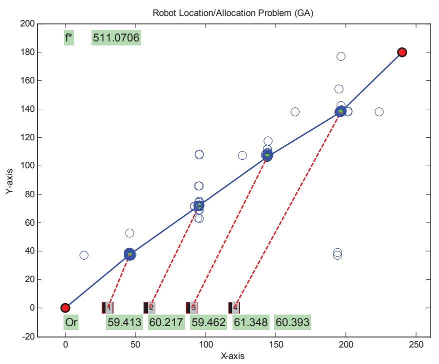
(b) At 1st iteration



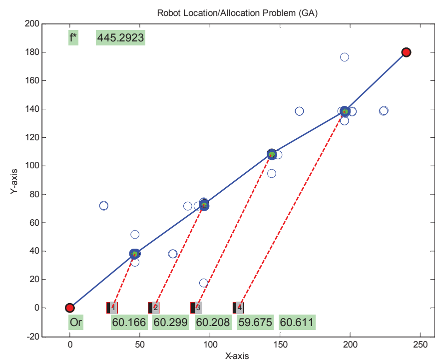
(c) At 10th iteration



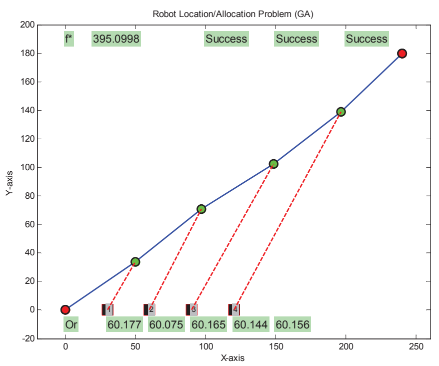
(d) At 30th iteration



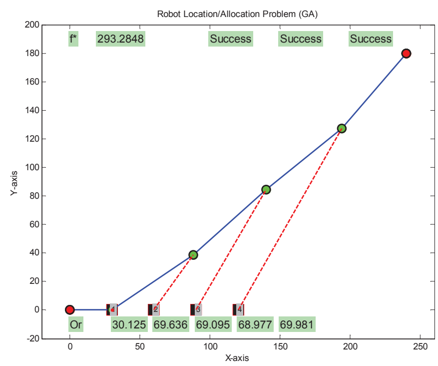
(e) At 90th iteration



(f) At 120th iteration



(g) GA, problems with $O_r = 60.2$



(h) GA, problems with $O_r = 70$

Figure 7: Results of the simulation on GA: (a) to (f) Transitions of gene's movement at different iterations; (g) Final results of the first scenario with $O_r = 60.2$; (h) Final results of the second scenario with $O_r = 70$. An example video showing a GA running is available at `https://youtu.be/mYSIWKahcJY`.

in a single iteration using two major components: a stochastic component and a deterministic component [41].

In PSO, a set of randomly generated particles searches the space of an objective function over a number of iterations, using a large amount of information sharing by all members of the swarm. The detailed steps of PSO that we exploited in this study can be found in [42]. Simply put, PSO requires three steps to meet the objectives of this research, as follows:

1) Initialization - The positions $\mathbf{x}_h^t$ and velocity $\mathbf{v}_h^t$ of N_{pop} particles are randomly generated using upper $\mathbf{x}^U$ and lower $\mathbf{x}^L$ bounds on the design variables as follows,

$$\mathbf{x}_h^0 = \mathbf{x}^L + r(\mathbf{x}^U - \mathbf{x}^L), \text{ and } \mathbf{v}_h^0 = 0, \ \forall h = 1, 2, \ldots, N_{pop} \tag{14}$$

where r represents a random number (i.e., $r = rand[0,1]$).

2) Velocity update -At each iteration, the velocities of all particles are updated as follows,

$$\mathbf{v}_h^{t+1} = w\mathbf{v}_h^t + \alpha r(\mathbf{g}^* - \mathbf{x}_h^t) + \beta r(\mathbf{x}_h^* - \mathbf{x}_h^t) \tag{15}$$

where $\mathbf{g}^*$and $\mathbf{x}_h^*$ are the position of the current global best position and each particle's best position, respectively; w is known as the inertial weight, α and β are acceleration constants and determine how much the particle is influenced by its best position.

3) Position update -This is the last step in each iteration, and the new position of each particle k can be updated by

$$\mathbf{x}_h^{t+1} = \mathbf{x}_h^t + \mathbf{v}_h^{t+1}. \tag{16}$$

In addition to the last two steps of velocity and position updates, fitness calculations using Eq. (1) are repeated until the total number of iterations reaches a fixed number N_{qa}, as the GA adopts this stopping criterion. For the *Allocation* problem, which we formulate as an integer problem, when $i = 1, 2, \ldots, n$ (i is the index of a decision vector $\mathbf{x}$), the values r are all rounded. This modification is then applied to step 1) and 2).

In order to validate the PSO for the network problem, we again implement the simulation with two scenarios, as in the previous section. Thus, every parameter for the simulation environment is the same as the GA implementation. Additionally, for the PSO, we set w to 0.5, α to 1.5, and β to 1.5, since we determined that they find the solutions well and provide the best convergence rate for the problems throughout this study.

The results are presented in Figure 8. Figure 8 (a) depicts the given problems, and Figures 8 (b) to (f) show the movements of particles marked with "x", at the different iterations of the first scenario. As shown in the figures, every particle converges as iterations increase. Also, the figures show that the PSO tries to solve the *Allocation* problem first and then focus on solving the *Location* problem, much like the GA. The final results of the first scenario are shown in Figure 8 (g). From this figure, it can be seen that the final nodes L_j form an almost-straight line that connects L_c to L_e, and every robot is allocated to a different node L_j. The latter result validates that the *Allocation* problem satisfies the equality constraint Eq. (3), and that the modification of the rounding process works well for an integer problem. Figure 8 (h) shows the result of the second scenario. From this figure, it can be seen that the final nodes L_j are located closer to the robots $\overline{R}_i$ than in the first scenario. With the naked eye, the results of the PSO simulations seem to indicate the feasibility of this optimization problem and are very similar to those of the GA.

5. Simulations and proof-of-concept study

Since heuristic optimization algorithms accompany a search task that may result in finding local minima and requiring a number of evaluations, we evaluate its completeness (i.e., finding the final solution), computational effort (i.e., the number of function evaluations), and consistency in finding the final solution, in this section.

5.1. Simulation Setup

For this further investigation, we employ the proposed method using GA[2]. The first test measures a success rate of finding an acceptable solution by the proposed algorithm, using 10-run trials for 12 test problems. Due to this large amount of test problems, we have decided not to impose the shortest path constraint when calculating an objective function for this simulation testing. In fact, we observed from our preliminary tests that employing Dijkstra's algorithm to find a shortest path takes about 0.02 sec for each run. This means that when there are two robots operated, GA takes more than 10,000 cost evaluations. Thus, it takes about 200 sec, i.e., longer than 3 minutes to compute this optimization problem, which might be too long to test all the trials and test problems.

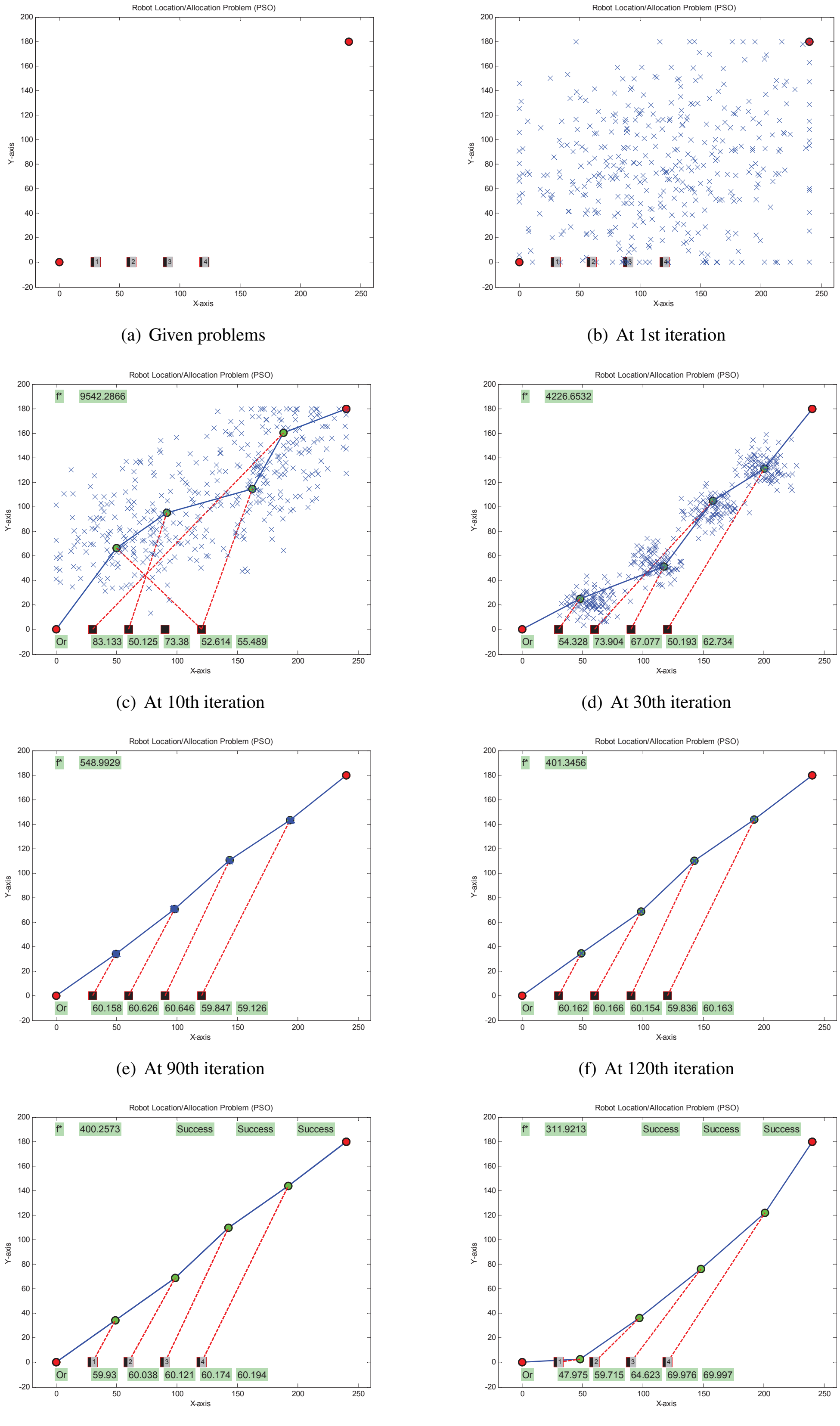
Note that the solutions may or may not be the global minimum. Thus, for convenience of evaluation, we will take into consideration whether or not the algorithm could find the solutions, if no constraints are being violated at the moment when the convergence criteria are met. The second test investigates the computational efforts of the proposed algorithm, using the same data and convergence criteria. The criteria are defined as follows,

$$\left| f(\mathbf{x}^t) - f(\mathbf{x}^{t-q}) \right| \leq \varepsilon. \tag{17}$$

Eq. (17) is satisfied when the maximum change in best fitness is smaller than the specified tolerance ϵ for a specified number of moves q. In this study, q is set to 10, and ϵ is set to 10^{-5}, for all test problems. In addition, the maximum number of function evaluations is included (i.e., the algorithm is stopped if t reaches the 150th iteration).

For a set of 12 test problems, we vary the number of robots n and the number of obstacles m. Two fixed end nodes are set to L_c = (0.0, 0.0), L_e = (240.0, 180.0), and the side constraint

[2]In fact, we also employed the PSO for this investigation during our preliminary study but it showed that results from the PSO are mostly similar to those from the GA and insignificant. Because of that, we do not include the results to this paper.



(a) Given problems

(b) At 1st iteration

(c) At 10th iteration

(d) At 30th iteration

(e) At 90th iteration

(f) At 120th iteration

(g) PSO, problems with $O_r = 60.2$

(h) PSO, problems with $O_r = 70$

Figure 8: Results of the simulation on PSO: (a) to (f) Transitions of particle's movement at different iterations; (g) Final results of the first scenario with $O_r = 60.2$; (h) Final results of the second scenario with $O_r = 70$. An example video showing a GA running is available at `https://youtu.be/YEBqGMHKJMo`.

Table 3: Nine problems with different settings

Problems	Number of Robots n	Number of Obstacles m	Operating range O_r
Problem 1	2	0	120.0
Problem 2	3	0	95.0
Problem 3	4	0	80.0
Problem 4	2	1	120.0
Problem 5	3	1	95.0
Problem 6	4	1	80.0
Problem 7	2	2	120.0
Problem 8	3	2	95.0
Problem 9	4	2	80.0
Problem 10	2	3	120.0
Problem 11	3	3	95.0
Problem 12	4	3	80.0

$x_{min} = 0.0$, $x_{max} = 240.0$, $y_{min} = 0.0$, and $y_{max} = 180.0$ for all the test problems. Also, all constraints we described in Eqs. (2) − (10) are included. Representative parameters are summarized in Table 3, and solution examples found by the GA in each test problem are depicted in Figure 9.

5.2. Results and Discussion

Results for the tests are graphically summarized in Figure 10 and Figure 11. Figure 10 (a) shows the success rate of finding a solution. As shown in the results, the proposed method mostly found the solution in the various environments. More specifically, it found the solutions 9 to 10 times out of each 10 trials in most cases (i.e, it shows a success ratio of higher than 90%). It is worth noting that there are a total of 7 failures occurred (from a total of 100 trials) in these tests. However, 4 of them almost met the constraints as we rounded off the nearest thousandth for the numbers of each constraint. Thus, only 3 manifest failures took place and actually they resulted from being trapped in local minimum.

Results on the computational effort of the proposed method are graphically summarized in Figure 10 (b) and (c). Figure 10 (b) shows the mean number of generations with 10-run trials in each problem that we conducted for the completeness test as shown 10 (a). As shown in this figure, the proposed method demonstrates that convergence occurs very fast. Most cases converge before reaching 50 generations and the graph shows that the smaller number of robots converges faster than the larger number of robots. This result is also shown in Figure 10 (c). Figure 10 (c) shows the mean number of generations with the standard deviation. This figure indicates that the number of robots could affect the computational effort (see P1 to P3 and P 10 to P12) while the number of obstacles rarely does.

Figure 11 shows the mean value of the final solution with standard deviation when the proposed method satisfies the stopping criterion in each problem. Note that values from failed trials are not included for this figure. As a result, variances of each graph show that the proposed method is consistent with finding the final solution. Specifically, when employing two robots, final objective values seem to be very consistent. This result and analysis are important in that this validates there is insignificant effect of the random number generator required for GA process.

From these tests on the proposed method's computational effort and consistency, we could observe that their variances become wider as the number of robots increases. Considering the number of variables in the optimization process increases as the number of robots increases (in fact, one increase in the number of robots introduces three more variables to be optimized), this increasing pattern could be expected. However, in order to make the proposed system fully scalable, we should also take into account this issue in the future. For this, we could consider implementing a parallel programming technique as described in [43].

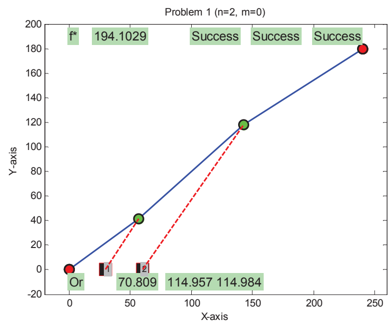
5.3. Proof-of-Concept Study

In this subsection, we present a set of proof-of-concept study to validate our hypothesis that eliminating intersections results in a simpler robot tracking system, as stated in Section 3.2.2.

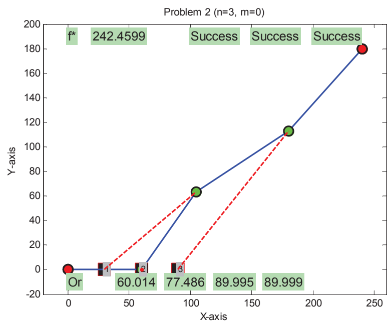
For the proof-of-concept study, we built miniature mobile robots that can be remotely manipulated, as shown in Figure 12 (a). We attached a particular color patch to each robot and installed a camera from the ceiling to identify the location and direction of the color patch to estimate the robot's current position in real time, as shown in Figure 12 (b). The computation for the *Location* and *Allocation* problem is computed offline and resultant location and allocation data are remotely sent to relevant robots to act online accordingly.

The first test was designed to analyze the difference in elapsed times between two cases: (1) when there is an intersection between the ground tracks of two robots, and (2) when there is no intersection. For this test, we initially placed two robots on the left side and designated two target locations that the two robots should move to on the right side. In this setup, two cases of an allocation were possible for each robot. The first allocation generated an intersection as shown in Figure 13 and the second allocation produced no intersection as shown in Figure 13. Figure 15 (a) show a summary of elapsed times. As can be seen when there were no intersections generated, the robots reached their goal positions faster than when there were intersections. These results could be expected because the robots must decrease velocity or stop until the other completely passes thru a region that is considered as a *warning area*, resulting in an increased total elapsed time. Therefore, we were able to validate our hypothesis that we can make a robot tracking system simpler by means of removing intersection points during the path planning stage.

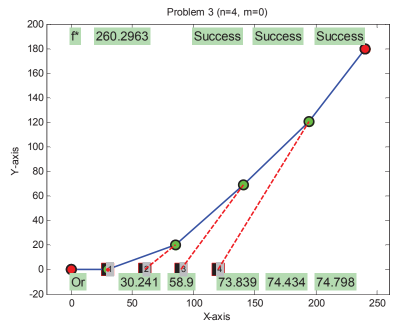
The second test was designed to analyze any differences in elapsed times between two cases: (1) when allocation is determined by considering the robots' distances to the nodes, and (2) when allocation is determined by considering the robots headings to the nodes. Resultant traces of the robots are depicted in Figure 14. The top figures show when the robots first considered their distance to the nodes, and the bottom figures show when the robots first considered their initial headings to the nodes. Figure 15 (b) shows a summary of elapsed times from these two different cases. As shown in the results, the two cases showed little difference in elapsed time. This signifies that to



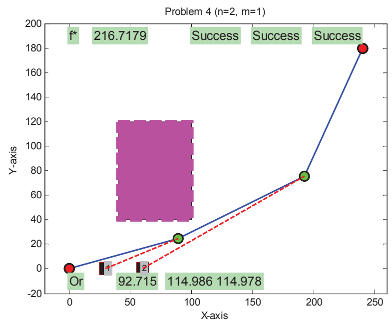
(a) Problem 1



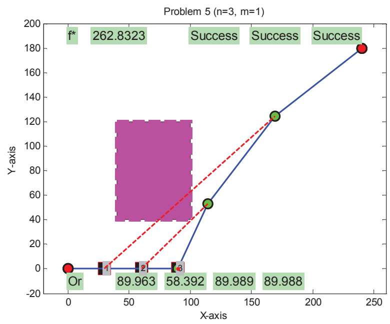
(b) Problem 2



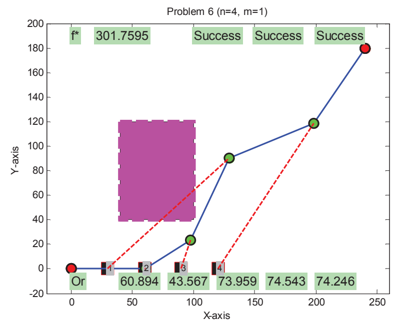
(c) Problem 3



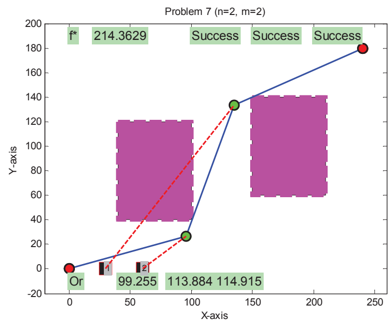
(d) Problem 4



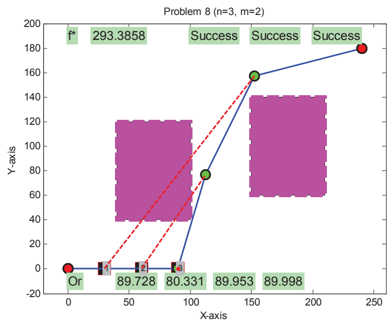
(e) Problem 5



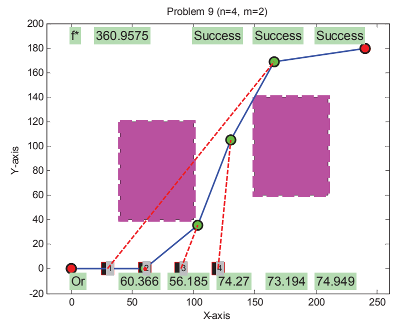
(f) Problem 6



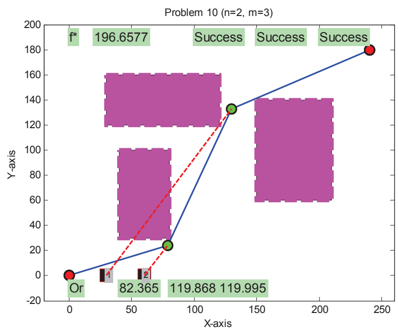
(g) Problem 7



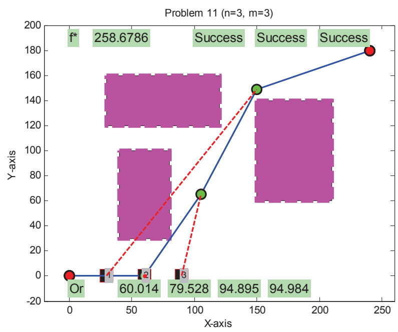
(h) Problem 8



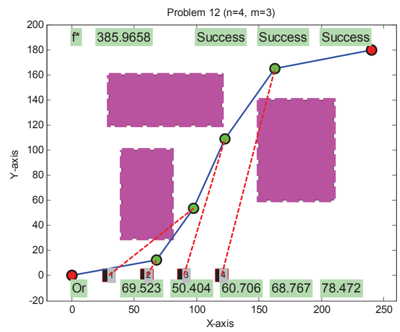
(i) Problem 9



(j) Problem 10

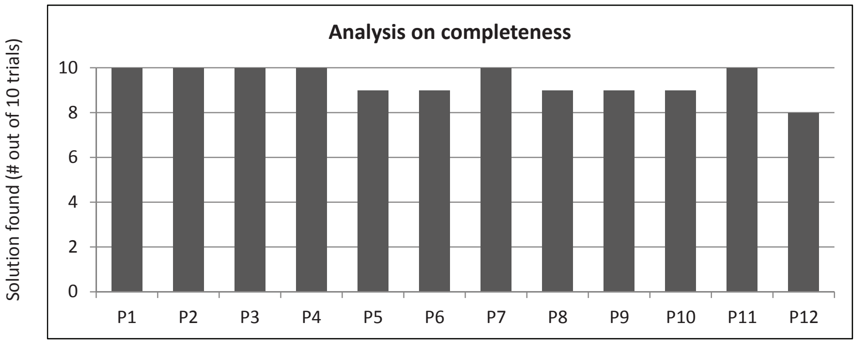


(k) Problem 11

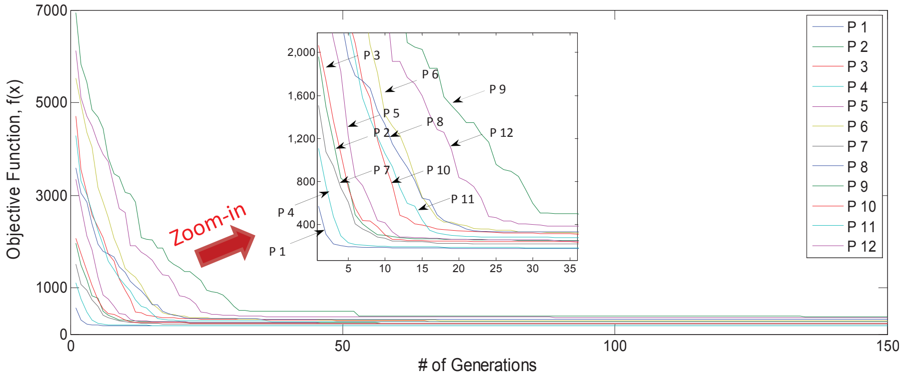


(l) Problem 12

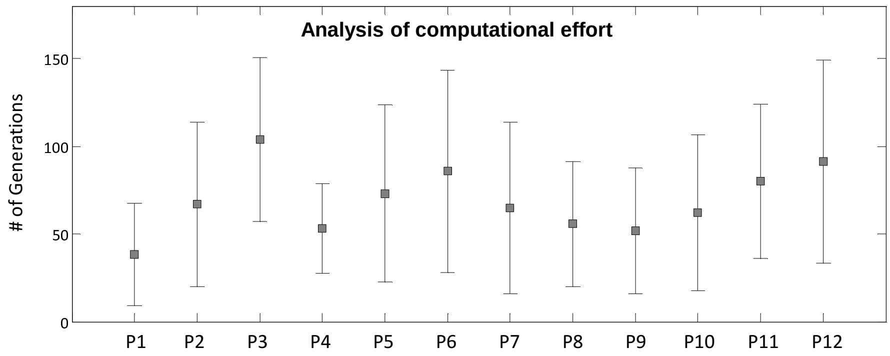
Figure 9: Solution examples to the 12 problems for further tests. The figures in the same column include the same number of robots n, and the figures in the same row include the same number of obstacles m.



(a) Completeness



(b) GA progress toward optimal solution



(c) Computational effort

Figure 10: Results on tests of the proposed method's completeness and computational effort in finding the solution: 10 trials were conducted for each problem.

minimize the total elapsed time for the robot task, the most important metric was distance from the initial location of robots to their goal position. Conversely, the initial heading of the robots is less important.

6. Field Experiments

6.1. Preparation for Experiments

To test the proposed methods, a prototype of the relay mobile robot system was developed and is shown in Figure 16. Every robot is homogeneous and equipped with the same components. Each robotic system is made up of the P3AT mobile robot, an embedded microprocessor with various sensors, AP (Access Point) with an omni-directional antenna, a wireless station with an omni-directional antenna, and a network switch, as shown in Figure 16 (b).

The P3AT, a compass and GPS sensors are connected by a serial connection to the embedded microprocessor that processes all required controls. For the network configuration, we set up the indirect point-to-point link with a combination of a station and AP available from Ubiquiti Networks Inc. This configuration acts like a very long wired cable and allows us to build a transparent Ethernet bridge between two end nodes wirelessly. One of the two end nodes, which is the left most node in Figure 16 (a), is a command center. The other end node, which is not shown in the figure, could be anything that has the capability to

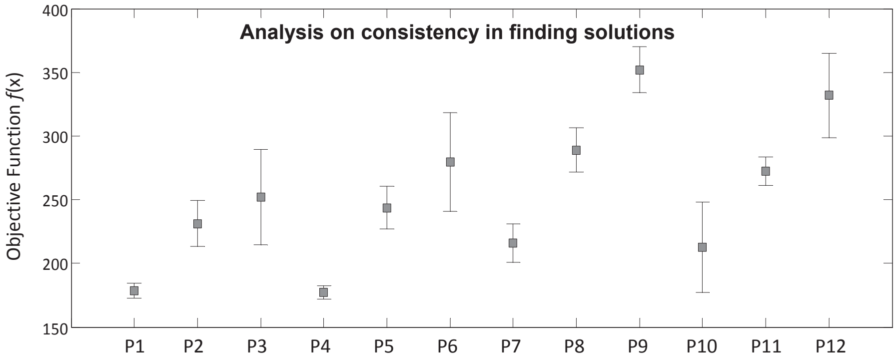
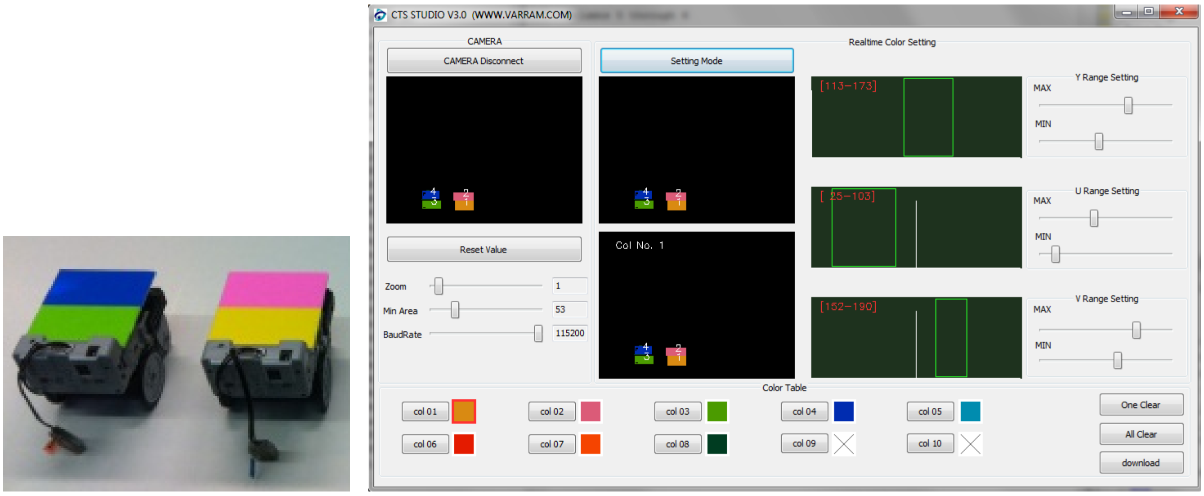


Figure 11: Results on tests of the proposed method's consistency with finding the solution: 10 trials were conducted for each problem.



(a) Miniature robots

(b) Robot control system

Figure 12: A proof-of-concept design.

carry a wireless station with an omni-directional antenna such as a robot or a human user. Every network device including antennas, an embedded microprocessor, and switches was given a static IP address for remote access and for receiving the desired position of each robot from the command center by numerically solving *Location* and *Allocation* problems. For example, "192.168.1.131" is assigned to the first relay robot and the position data are transfered to the robot through the IP address.

6.2. Experiments

In order to validate the proposed system, we conducted extensive field tests using two different sets: 1) with three robots and 2) with two robots. The tests were conducted at the Cumberland Park in West Lafayette, Indiana USA, which is a wide open site and the direct distance between the two end nodes was approximately 240 meters. For the tests, robots were initially located around one of the end nodes, i.e., the command center and for the other end node, we installed a laptop running the "iperf" Linux command for a data throughput test while relaying robots move away from the command center to their destinations. This was done to reinforce the assumption that the calculated locations have a direct correlation to the best signal for a data link connection. To test this, a laptop was set to a server mode, and a laptop on the command center side was set to a client mode. A small amount of data was transferred through the autonomously created link and a measurement of the time to transfer rate was performed by iperf. The resulting measurement gives an accurate available throughput for the established link.

The first test was done using three mobile robots with two virtual obstacles as shown in Figure 17 (a). Given two endpoints (red circles in the figure) and map information such as the physical location of obstacles (magenta rectangles in the figure), the *Location* and *Allocation* problem was solved (the three green circles indicate the calculated locations where the relay robots should reach) at the command center side, and calculated data were remotely sent to each robot. Upon receiving the position data every robot started moving to the given way points generated with the Dijkstra's algorithm to reach their destinations using a set of sensors such as GPS and a compass sensor. Actual traces of the robots are depicted in Figure 17 (a), and actual locations (relative travels) of the robots along with the elapsed times are depicted in the top three graphs in Figure 17

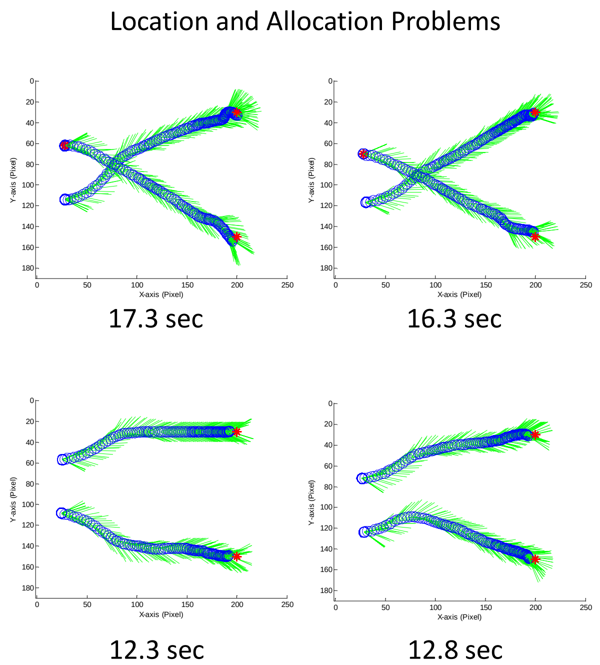


Figure 13: Trace of robots from proof-of-concept tests on intersection (top) vs. no intersection (bottom).

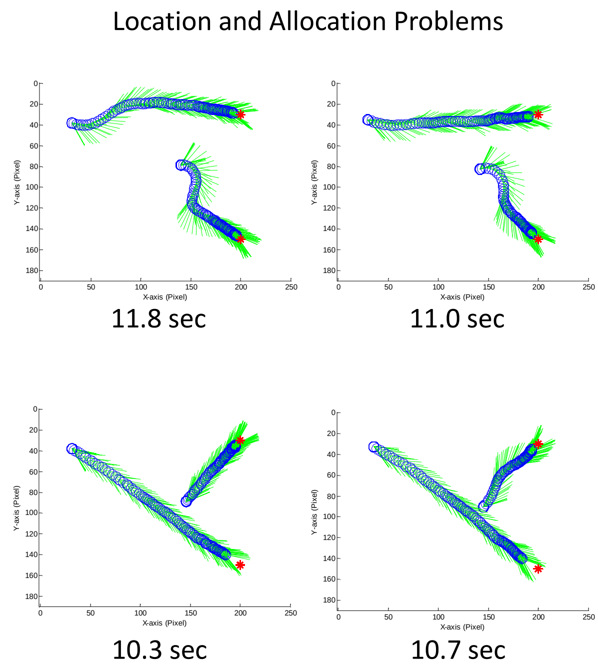
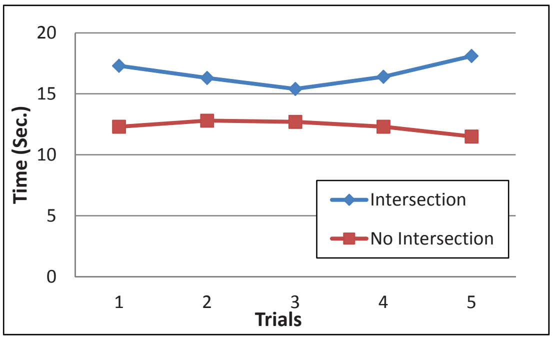


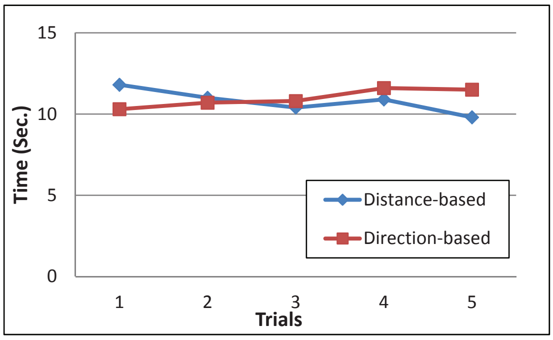
Figure 14: Trace of robots from proof-of-concept tests on distance-based (top) vs. heading-based (bottom).

(b). The top three graphs show that the robot 3 reached the destination first, followed by the robot 1 and the robot 2. As clearly shown in the figure, the *Location* and *Allocation* problem was successfully solved, and every robots was able to explore and reach their destinations safely and successfully. Results of the network performance are shown in the bottom of Figure 17 (b). Notably, an end-to-end communication started to be established after 150 seconds lapse, which is really quick, considering the robots being operated with a slow speed and in the large environments. It is not a surprise that the communication could not be established at all for the first 150 seconds since some robots were far yet to connect their neighboring nodes including the two end nodes. On the other hand, the data throughput noticeably increased as the robots were approaching their destinations, i.e., an optimal location for the end-to-end communication. The data throughput recorded the highest value (7.88 Mbps) when the robot all reached their destinations, and therefore this clearly shows that our proposed method is validated.

The second test was done using two mobile robots with two virtual obstacles in the same environment as shown in Figure 18 (a). Only difference from the previous test was the number of the relay robots for the problem. This test was designed to show the scalability of the scheme in this research. As a result, the *Location* and *Allocation* problem was again successfully solved with a longer operating range O_r, and every robots was able to reach their destinations safely as shown in Figure 18 (a). The top two graphs show that the robot 2 reached the destination first, followed by the robot 1. Notably, an initial end-to-end communication was established in a short time (i.e., it only took 120 seconds), and as similar to the first test, the communication could not be established at all for the first 120 seconds as shown



(a) Intersection vs. no intersection

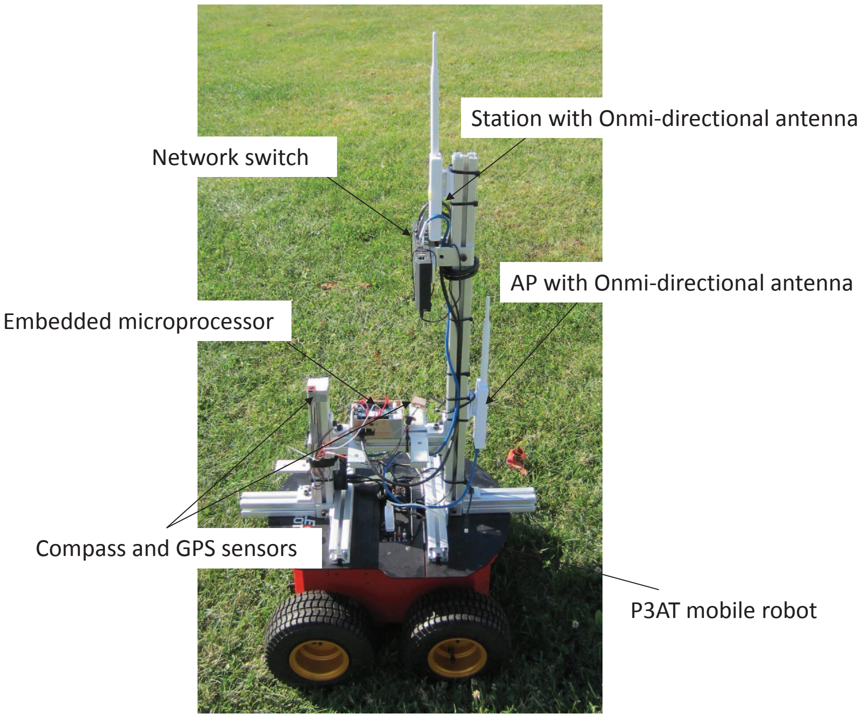


(b) Distance-based vs. heading-based

Figure 15: Elapsed times from five different trials.

in Figure 17 (b). However, as the robots were approaching their destinations, the data throughput increased and when the robot all reached their destinations, the data throughput was the high-

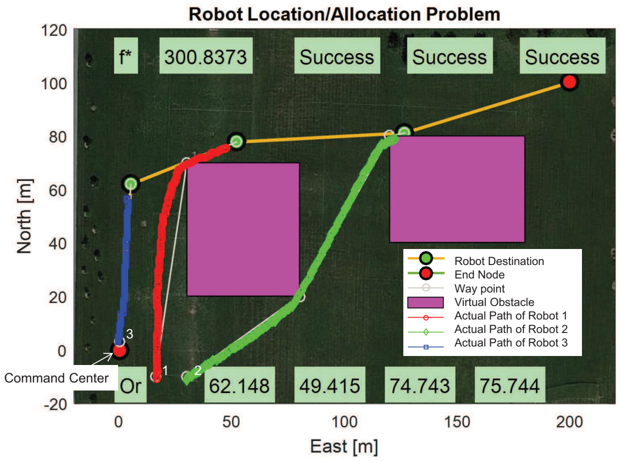
(a) A complete relay robot team



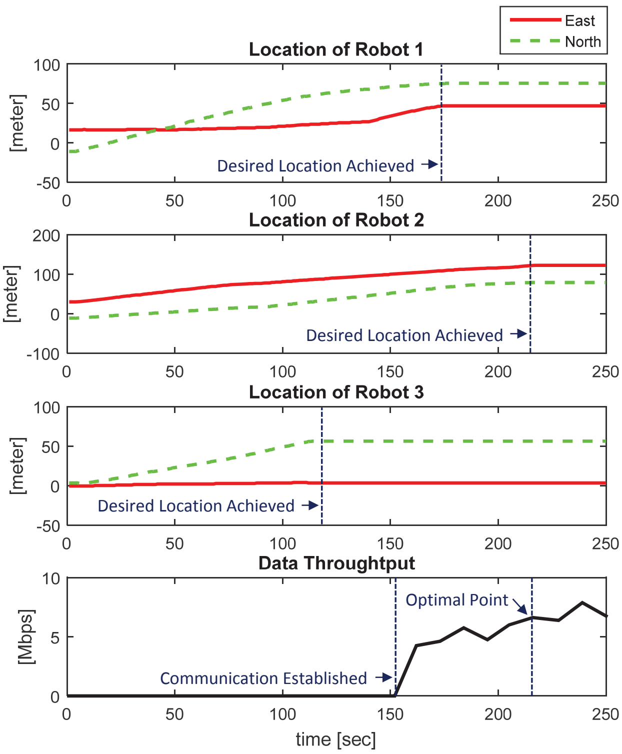
(b) A robot with network components

Figure 16: A robotic relay system for the establishment of an end-to-end communication.

est (6.38 Mbps). Because only two robots were used in the environment, the final throughput was a bit lower than when three robots were employed. While this shows that the use of the three relay robots would be a better option for a better network quality in this environment, the use of the two robots would be a better option for a quicker establishment of the communication link. The bottom graph in Figure 17 (b) shows that there are some throughput drops after optimal points were achieved, but this was caused by a human intervention who was getting closer to the second mobile robot and acted as a physical obstacle to

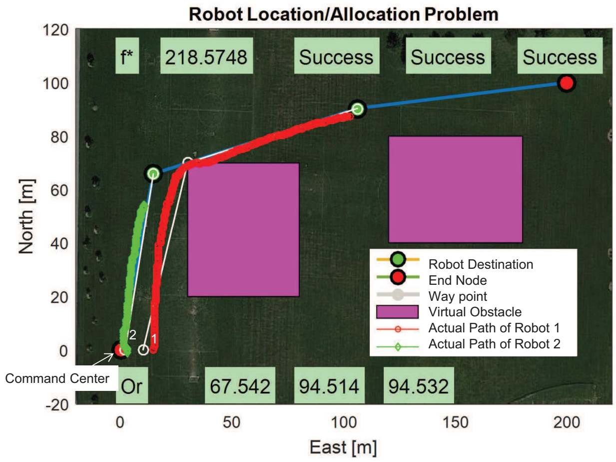


(a) Problem set

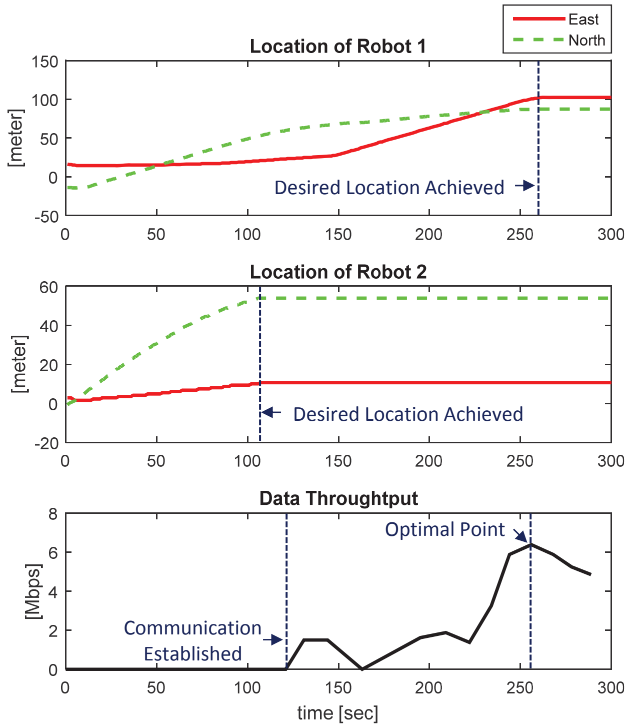


(b) Actual locations of robots and data throughput

Figure 17: Experiments with three robots. (a) shows that solutions (green dots) to the *Location* and *Allocation* problem, given environments with two end points (red dots) and physical obstacles (magenta rectangles), and actual paths of robots (red lines with a circle green lines with a diamond, and blue lines with a square). (b) shows that robots' actual relative movements and data throughput along with the elapsed time. A video demonstrating this field experiment can be found at `https://youtu.be/o_25ZLFcNeA`



(a) Problem set



(b) Actual locations of robots and data throughput

Figure 18: Experiments with two robots. (a) shows that solutions (green dots) to the *Location* and *Allocation* problem, given environments with two end points (red dots) and physical obstacles (purple rectangles), and actual paths of robots (green lines with a diamond and blue lines with a circle). (b) shows that robots' actual relative movements and data throughput along with the elapsed time. A video demonstrating this field experiment can be found at `https://youtu.be/8imsFu0-4Wo`

the established wireless communication (you can check this out the video). As a result, this test also clearly demonstrates that our proposed method is feasible and effective in a real world situation.

7. Conclusions and Future Works

In this paper, we set a goal of minimizing the path required for the deployment of networked robots in order to relay two given end nodes and therefore create an end-to-end communication network. To achieve this goal, we addressed the fundamental problem of finding optimal locations and subsequent robot allocation. We present two optimization techniques: GA and PSO, and we described constraints on the problem, by considering the propagation of radio signals, infeasible robot locations, and intersections between robot paths. Our simulation testing results validate that the proposed methods are able to find the acceptable solution and that they are robust and efficient. In addition, proof-of-concept study and field experiments demonstrated the effectiveness of the proposed concept and algorithms.

This research mainly aims at introducing a way of applying two representative evolutionary heuristic algorithms to the robotic network problem, and it is shown that they both are very feasible. For potential future works, investigating their convergence and efficiency would allow determining the more suitable algorithm for this problem.

It is worth noting that although this paper supposes all robots would carry the same wireless devices having the same operating ranges, the optimization problem we formulated allows using robots carrying different wireless devices having different operating ranges. This flexibility would be greatly beneficial especially when robots are heterogeneous cooperating together to achieve their goals.

In addition, our proposed method can be greatly improved if we exploit directional antennas for wireless communication [35, 36]. To do so, we will solve another optimization problem: finding the best orientations of the directional antennas, carried by the robots or placed at the locations of the end nodes L_c and L_e, in order to maximize the Received Signal Strength Indication (RSSI). Combining and solving these two optimization problems at the same time will enable the formation of the long-distance coverage network.

Acknowledgements

The authors thank the many members of the research team who assisted with different aspects of this work; notably Jun Han Bae and Seongha Park.

References

[1] National Emergency Communications Plan, U.S. Department of Homeland Security, 2014.

[2] H. G. Nguyen, N. Pezeshkian, A. Gupta, N. Farrington, Maintaining communications link for a robot operating in a hazardous environment, in: ANS 10th Int. Conf. on Robotics and Remote Systems for Hazardous Environments, 2004.

[3] B.-C. Min, E. Matson, A. Smith, J. Dietz, Using directional antennas as sensors to assist fire-fighting robots in large scale fires, in: 2014 IEEE Sensors Applications Symposium (SAS), 2014.

[4] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, M. Zadimoghaddam, Minimizing movement, ACM Trans. Algorithms 5 (3) (2009) 30:1–30:30.

[5] J. Reich, V. Misra, D. Rubenstein, G. Zussman, Connectivity maintenance in mobile wireless networks via constrained mobility, Selected Areas in Communications, IEEE Journal on 30 (5) (2012) 935–950.

[6] H. G. Nguyen, H. R. Everett, N. Manouk, A. Verma, Autonomous mobile communication relays test, in: Proc. SPIE, Vol. 4715, 2002, pp. 50–57.

[7] C. Q. Nguyen, B. C. Min, E. T. Matson, A. H. Smith, J. E. Dietz, D. Kim, Using mobile robots to establish mobile wireless mesh networks and increase network throughput, International Journal of Distributed Sensor Networks 2012.

[8] G. Tuna, V. C. Gungor, K. Gulez, An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters, Ad Hoc Networks 13 (2014) 54–68.

[9] B.-C. Min, E. T. Matson, Robotic follower system using bearing-only tracking with directional antennas, in: Robot Intelligence Technology and Applications 2, Springer, 2014, pp. 37–58.

[10] K.-H. Kim, K. Shin, D. Niculescu, Mobile autonomous router system for dynamic (re)formation of wireless relay networks, Mobile Computing, IEEE Transactions on 12 (9) (2013) 1828–1841.

[11] N. Bezzo, R. Fierro, Tethering of mobile router networks, in: American Control Conference (ACC), 2010, 2010, pp. 6828–6833.

[12] Y. Pei, M. W. Mutka, Steiner traveler: Relay deployment for remote sensing in heterogeneous multi-robot exploration, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, 2012, pp. 1551–1556.

[13] Y. Yan, Y. Mostofi, Robotic router formation in realistic communication environments, Robotics, IEEE Transactions on 28 (4) (2012) 810–827.

[14] O. Tekdas, Y. Kumar, V. Isler, R. Janardan, Building a communication bridge with mobile hubs, IEEE Transactions on Automation Science and Engineering 9 (1) (2012) 171–176.

[15] C. Dixon, E. Frew, Maintaining optimal communication chains in robotic sensor networks using mobility control, Mobile Networks and Applications 14 (3) (2009) 281–291.

[16] C. Costanzo, V. Loscri, E. Natalizio, T. Razafindralambo, Nodes self-deployment for coverage maximization in mobile robot networks using an evolving neural network, Computer Communications 35 (9) (2012) 1047–1055.

[17] V. Loscri, E. Natalizio, T. Razafindralambo, N. Mitton, Distributed algorithm to improve coverage for mobile swarms of sensors, in: Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on, 2013, pp. 292–294.

[18] V. Loscrí, E. Natalizio, F. Guerriero, N. Mitton, Efficient coverage for grid-based mobile wireless sensor networks, in: Proceedings of the 11th ACM symposium on Performance evaluation of wireless ad hoc, sensor, & ubiquitous networks, ACM, 2014, pp. 53–60.

[19] V. Loscri, E. Natalizio, N. Mitton, Performance evaluation of novel distributed coverage techniques for swarms of flying robots, in: Wireless Communications and Networking Conference (WCNC), 2014 IEEE, IEEE, 2014, pp. 3278–3283.

[20] A. Klose, A. Drexl, Facility location models for distribution system design, European Journal of Operational Research 162 (1) (2005) 4–29.

[21] M. Gen, R. Cheng, L. Lin, Network Models and Optimization: Multiobjective Genetic Algorithm Approach, 1st Edition, Springer Publishing Company, Incorporated, 2008.

[22] T.-H. Wu, C. Low, J.-W. Bai, Heuristic solutions to multi-depot location-routing problems, Computers & Operations Research 29 (10) (2002) 1393–1415.

[23] M. Wen, K. Iwamura, Fuzzy facility location-allocation problem under the hurwicz criterion, European Journal of Operational Research 184 (2) (2008) 627–635.

[24] A. R. Guner, M. Sevkli, A discrete particle swarm optimization algorithm for uncapacitated facility location problem, Journal of Artificial Evolution and Applications 2008 (2008) 10.

[25] I. Stojmenovic, X. Lin, Power-aware localized routing in wireless networks, Parallel and Distributed Systems, IEEE Transactions on 12 (11) (2001) 1122–1133.

[26] E. Natalizio, V. Loscri, E. Viterbo, Optimal placement of wireless nodes for maximizing path lifetime, Communications Letters, IEEE 12 (5) (2008) 362–364.

[27] K. Miranda, E. Natalizio, T. Razafindralambo, Adaptive deployment scheme for mobile relays in substitution networks, International Journal of Distributed Sensor Networks 2012.

[28] F. Guerriero, A. Violi, E. Natalizio, V. Loscri, C. Costanzo, Modelling and solving optimal placement problems in wireless sensor networks, Applied Mathematical Modelling 35 (1) (2011) 230–241.

[29] E. Natalizio, V. Loscri, F. Guerriero, A. Violi, Energy spaced placement for bidirectional data flows in wireless sensor network, Communications Letters, IEEE 13 (1) (2009) 22–24.

[30] V. Loscrí, E. Natalizio, C. Costanzo, Simulations of the impact of controlled mobility for routing protocols, EURASIP J. Wirel. Commun. Netw. 2010 (2010) 7:1–7:12.

URL http://dx.doi.org/10.1155/2010/315381

[31] D. K. Goldenberg, J. Lin, A. S. Morse, B. E. Rosen, Y. R. Yang, Towards mobility as a network control primitive, in: Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '04, 2004, pp. 163–174.
URL http://doi.acm.org/10.1145/989459.989481

[32] C. Nguyen, B. Leong, E. Matson, A. Smith, J. Wachs, AWARE: autonomous wireless agent robotic exchange, Intelligent Robotics and Applications (2010) 276–287.

[33] A. Smith, E. T. Matson, Use of antennas as sensors to discover signals to form mobile broadband networks, in: Sensors, 2009 IEEE, 2009, pp. 581–584.

[34] A. O. Tekdas, Improving communication in networked systems using mobile robots, Ph.D. thesis, University of Minnesota, Minneapolis, MN, USA (2011).

[35] B.-C. Min, J. Lewis, D. Schrader, E. Matson, A. Smith, Self-orientation of directional antennas, assisted by mobile robots, for receiving the best wireless signal strength, in: 2012 IEEE Sensors Applications Symposium (SAS), 2012.

[36] B.-C. Min, J. Lewis, E. T. Matson, A. H. Smith, Heuristic optimization techniques for self-orientation of directional antennas in long-distance point-to-point broadband networks, Ad Hoc Networks 11 (8) (2013) 2252–2263.

[37] B. Brggemann, M. Brunner, D. Schulz, Spatially constrained coordinated navigation for a multi-robot system, Ad Hoc Networks 11 (7) (2013) 1919 – 1930.

[38] M. A. Vieira, R. Govindan, G. S. Sukhatme, An autonomous wireless networked robotics system for backbone deployment in highly-obstructed environments, Ad Hoc Networks 11 (7) (2013) 1963 – 1974.

[39] S. Skiena, Dijkstra's algorithm, Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley (1990) 225–227.

[40] S. M. LaValle, Planning Algorithms, Cambridge University Press, New York, NY, USA, 2006.

[41] M. R. Amini, Geolocation estimation for mobile users in umts by particle swarm optimization., Proceedings of World Academy of Science: Engineering & Technology 50.

[42] R. Hassan, B. Cohanim, O. De Weck, G. Venter, A comparison of particle swarm optimization and the genetic algorithm, in: Proceedings of the 1st AIAA Multidisciplinary Design Optimization Specialist Conference, 2005.

[43] T. L. James, R. Barkhi, J. D. Johnson, Platform impact on performance of parallel genetic algorithms: Design and implementation considerations, Engineering Applications of Artificial Intelligence 19 (8) (2006) 843–856.