Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2015) 23: 769 – 788
© TÜBİTAK
doi:10.3906/elk-1302-136

Moving as a whole: multirobot traveling problem constrained by connectivity

Yun WANG[*], **Cheng HU**
Key Lab of Network and Information Integration, Ministry of Education School of Computer Science and Engineering, Southeast University, Nanjing, P.R. China

Received: 19.02.2013 • **Accepted/Published Online:** 25.05.2013 • **Printed:** 30.04.2015

Abstract: A multirobot system provides many advantages over a single robot. In certain situations, robots need to maintain global connectivity while proceeding through tasks such as traveling to spots of interest in an area. This paper formulates the problem of multiple robots traveling while constrained by connectivity and analyzes the limits of increase in total traveling distance (TTD) caused by connectivity constraints. A connection condition is proposed and proven, which can be used to direct the design of solutions. Two algorithms satisfying the connection condition, connected nearest neighbor (CNN) and bold Lin–Kernighan heuristic (B-LKH), are proposed to solve this problem, which consider the connectivity constraint in planning paths. Simulations are designed to investigate the influence of important parameters, and comprehensive comparisons among algorithms are also conducted. The results show that CNN and B-LKH significantly outperform previous systems with respect to TTD.

Key words: Multiple traveling salesmen problem, multirobot system, global connectivity, nearest neighbor, bold Lin–Kernighan heuristic

1. Introduction

In recent years, research on control of multirobot systems (MRSs) has gained much attention due to the advantages provided by a robot network compared to a single robot. A MRS is composed of a network of robots that interact with each other to achieve a common goal. It is more efficient, redundant, and fault-tolerant than a single robot when completing a job.

In situations where static infrastructures are not available, an MRS can be quickly deployed and can provide sensing and actuating abilities. Thus, it can be used in a variety of missions such as surveillance in hostile or severe environments (i.e. border patrol, disaster sites, and areas contaminated with biological, chemical, or even nuclear wastes), large-area environmental monitoring (i.e. air quality monitoring, forest monitoring, and wild-animal monitoring), etc. Generally, in these missions, the area of interest cannot be covered by robots, which gives rise to the problem of traveling to interested spots (ISs).

This paper presents and solves a multirobot traveling problem constrained by connectivity (MRTPCC). Informally speaking, given several robots and predetermined positions (i.e. ISs) on a map, the problem is to assign travel paths to the robots for them to traverse all the ISs while making sure that they can communicate with one another via wireless radio all the time. The objective of the study is to minimize the total traveling distance (TTD) of the robot team. Although this problem seems very similar to the multiple traveling salesmen problem (MTSP), the connectivity constraint makes it more difficult to solve. First, some robots will have to

[*]Correspondence: yunwang@seu.edu.cn

move to non-ISs to keep the robot network connected, so it significantly expands the search scope and thus makes traditional heuristic searching methods impractical. Second, the robots have to be connected instantly, which means that the solutions have to take temporal factors into consideration, while traditional methods are merely time-sensitive. Last but not least, network partitioning has to be prevented, which increases the communication and computational burdens of the solution. Note that the MRTPCC is the same as the MTSP when the communication range of the robots is infinite. The MTSP is known to be NP-complete, and the MRTPCC is readily proven to be NP-complete.

To solve this problem, the influence of the connectivity constraint is theoretically analyzed, and a connection condition is proposed and proven. Under the guidance of the connection condition, a nearest neighbor-based method called connected nearest neighbor (CNN) and a bold Hamilton circle-based method called the bold Lin–Kernighan heuristic (B-LKH) are proposed. Both of these algorithms ensure global connectivity in rounds. In each round, robots reach as many ISs as possible. The difference of these algorithms lies in how they select goal positions for each robot in one round. CNN is stateless and lightweight and uses a nearest-neighbor criterion, while B-LKH makes selections based on a bold Hamilton circle. With global optimization, it generates shorter paths. The details of these two algorithms are illustrated in Section 5.

This work is motivated by a practical monitor mission, in which a network of robots equipped with cameras patrol a school and send the videos back to a sink computer through Wi-Fi. The data rate generated by the camera is up to 2 Mbps; however, the storage space on each robot is only 256 MB, which means that if a robot loses its connection to the sink computer for a period of about 20 min, the data will be lost. In an outdoor environment, the communication range of Wi-Fi is typically 20–30 m, and thus data loss is constant if the paths of robots are not properly scheduled. In a multirobot search and rescue system [1–3], this concern becomes more serious because losing connection may cause unnecessary casualties. Maintaining connectivity of a robot team is also crucial in applications of persistent surveillance, joint exploration, and mapping, where robots must have the ability to align themselves along the boundaries of complex shapes in two dimensions while ensuring the successful transmission of critical data [4–7]. Other applications requiring connectivity preservation include rendezvous and formation control [8–11], wireless distributed computing [12], etc.

The contributions of this paper are summarized as follows: 1) The MRTPCC, a practical variation of the MTSP, is formulated. 2) The influence of maintaining global connectivity is theoretically analyzed, and a connection condition is proposed and proven, which can be used to direct the design of solutions to MRTPCC. 3) Two algorithms are proposed to solve the MRTPCC. CNN is lightweight and robust to environmental dynamics, while B-LKH generates solutions with less TTD as the cost of higher computational complexity. (4) Simulations are designed to investigate the influence of the amount of robots and ISs, communication range, and IS distributions. Comparisons of the algorithms are also made.

The remainder of this paper is organized as follows: in the next section, related work on multirobot systems is reviewed. Section 3 gives the system model and problem formulation. Section 4 analyzes the influence of keeping connectivity and proposes and proves the connection condition. The CNN and B-LKH algorithms are proposed in Section 5. Simulations are conducted in Section 6. Section 7 concludes the paper.

2. Related work

Traversing ISs is part of a patrol planning problem, or a persistent surveillance planning problem [13], based on whether the visits should be regular. Generally, the solutions to this problem can be divided into 2 classes. Stump et al. [14,15] considered a persistent surveillance problem in which ISs are periodically visited over a long period of time. They approximated the problem to a vehicle routing problem with time windows. By properly modifying the original problem, they got a mixed integer linear problem formulation, which was further decomposed into subproblems by using Lagrange multipliers to relax the constraint that each site can only be visited by one vehicle. The proposed method is centralized and robust to mission dynamics. However, it has to compute for each time window, and it does not consider connectivity of the robots.

The other viewpoint is to take it as a MTSP. Trevai et al. [16] considered multirobot surveillance in unknown environments, where the boundary and obstacles were not known. By having robots sweep through a bounded region to perform a search requiring maximum coverage, iterative surveillance paths were planned, aiming to minimize the distance of the cyclic path. A simple but ineffective method is to find a TSP cyclic path along all the observation points and then make the robot repeat this travel cycle over and over. Khamis et al. [17] solved the problem of task allocation in mobile surveillance systems by applying a market-based approach. The shortest sequence planning (SSP) algorithm was used to find the minimum cost path for each robot given the ISs. The solution needed to perform a centralized SSP algorithm first, and then adjust the scheme in a distributed way. Furthermore, neither [16] nor [17] took connectivity into consideration.

The MTSP has been widely studied in applications such as school bus routing, print press scheduling, crew scheduling, etc. [18]. Current solutions include exact solutions and heuristic solutions. Bellmore et al. [19] converted the asymmetric MTSP with m salesmen and n nodes into a standard asymmetric TSP with $n+m-1$ nodes. Rao [20] gave the transformation of the symmetric MTSP into a standard symmetric TSP. Xu et al. [21] developed a 3/2-approximation algorithm for the multidepot MTSP, which partially filled the gap between the existing best approximation ratios for the TSP and for the multidepot MTSP in the literature. Therefore, the resulting problem can be solved by any algorithm devised for the TSP. These transformations make copies of nodes and generate an expanded graph that increases the problem scale and complexity. The TSP has been intensively studied for decades [22], and the Lin–Kernighan heuristic (LKH) [23,24] is generally considered to be one of the most effective solutions for generating optimal or near-optimal solutions for the symmetric traveling salesman problem. However, to the best of our knowledge, none of the algorithms mentioned above have considered the connectivity among the salesmen.

In many multirobot applications, maintaining connectivity is essential to their performance and the results. Sugiyama et al. [1] investigated a multirobot rescue system consisting of a base station and autonomous mobile robots. These robots coordinately formed a chain network to explore a disaster area and sent the video back to the base station. The connectivity of this chain network is essential for reconnaissance. In this work, the robots were divided into relay robots and search robots, depending on whether they were at the end of the chain network. Relay robots kept their master search robot connected to the fixed base station. Arkin et al. [5] investigated how a team of robotic agents self-organized for the exploration of a building, subject to the constraint of maintaining line-of-sight communications. Based on the information available, 3 different behavioral strategies were developed and tested. Two behavioral assemblages were involved: wander-avoid-past and living-in-past. Vazquez et al. [6] presented a behavior-based architecture for multirobot exploration and mapping. In this architecture, if the signal strength among the robots on the bridge of the robot network fell below a threshold, the achieve-connectivity action was executed. Otherwise, the robots moved to their goals in a distributed manner while avoiding collisions. However, this strategy cannot guarantee global connectivity when one set of robots loses connection with another set simultaneously. Hsieh et al. [4] implemented a neighbor-based method where robots stayed connected to certain neighbors. If the link quality among the robots fell below a certain threshold, they moved to these neighbors. Robots moved to their goals when the link quality was at an acceptable level. Anderson [7] and Derbakova [25] considered the repair of a broken robot network due to robot failures. Robots wander to reach their goals instead of moving cooperatively, which increases the TTD to keep connectivity or avoid collision with each other. Ponda et al. [26] addressed network connectivity issues in a more generalized task allocation application. The goal was to find a conflict-free allocation of tasks, valid in certain time windows, that maximized the global score. They improved the consensus-based bundle algorithm by explicitly considering the connectivity constraints in the planning process. However, these algorithms do not consider the traveling distance when planning the task allocation. A higher score does not ensure less TTD.

Preserving global connectivity in MRSs was also intensively studied by Ji and Egerstedt [8], Dimarogonas and Kyriakopoulos [9], and Sabattini et al. [10], where algebraic graph theoretic tools were used. Each robot was assigned a control law, in which parameters were to be adaptively and locally estimated. Obeying the control law, robots ensured that they kept connected to others, and thus applications like rendezvous and formation control while preserving connectivity were achieved. Esin et al. [11] presented a novel method for the formation control of a group of nonholonomic mobile robots using implicit and parametric descriptions of the desired formation shape. The formation control strategy employs implicit polynomial representations to generate potential fields for achieving the desired formation and the elliptical Fourier descriptors to maintain the formation once achieved. However, these studies just consider one goal position for each robot, leaving the allocation of all ISs unmentioned.

Compared to previous work, the algorithms proposed in this paper explicitly consider connectivity in planning the paths. First, it is more efficient and effective than methods that try to fix connectivity only when links are nearly broken. Second, to ensure connectivity of the plans, a connection condition is proposed and proved. CNN and B-LKH modify the standard nearest neighbor method and LKH to satisfy the connection condition. In B-LKH, the concept of nodes is expanded by applying a set-covering method. Since the proposed algorithms meet the connection condition in each step, they are proven to be correct. The details of the algorithms are presented in Section 5.

3. System model and problem formulation

Consider a planar rectangular area with no obstacles, where N robots are initially deployed at a service station (SS). These robots are equipped with a wireless communication module, a computing module, a GPS module, and a moving module. Each robot communicates with robots within distance r directly and with those outside its communication range in an ad hoc way if relay robots exist.

Note that r is the effective communication distance rather than the maximum communication range $r_{\max}$, because the communication model is actually not an ideal circle [27]. Additionally, each robot occupies a circular space of radius r_0. Robots can move at a maximum speed V, and turning time is neglected.

This paper considers the traveling problem in flat open areas, where obstacles are few in number and easy for robots to cross over. Since generally ISs are far away from each other, collision among robots rarely occurs. Even if it happens, the robots will execute a simple avoidance procedure automatically, so it is reasonable to omit obstacles and robot size r_0 in the model.

In a travel mission, there are M ISs to be traversed, whose coordinates are predetermined. The N robots start from the SS, traverse all the ISs, and then return to the SS. Generally, $N < M$ holds. Both ISs and robots are regarded as points on the plane, which are characterized by their coordinates. By constantly exchanging information with others, robots are available to obtain global knowledge about the mission and the map.

Formally speaking, ISs are the positions on the plane, which have to be visited by a robot at least once, and any other positions on the plane are called auxiliary spots (ASs). Robot R_i is called connected with R_j if 1) $D(R_i, R_j) \leq r$, where $D(R_i, R_j)$ calculates the distance of R_i and R_j; or 2), there exists a robot R_k, such that $D(R_i, R_k) \leq r$ and R_k is connected with R_j. Denote $C(R_i, R_j) = 1$ if R_i is connected with R_j, and C(R_i, R_j) $= 0$ otherwise. A robot network S is a set of robots such that for $\forall R_i$, R_j in the set, $C(R_i, R_j) = 1$. A robot R_i is said to be connected with a robot network S if $\exists R_j \in S$, such that C(R_i, R_j) $= 1$.

The MRTPCC is formulated as follows. Given M ISs, i.e. IS_1, IS_2, ..., IS_M, and N robots, i.e. R_1, R_2, ..., R_N, try to designate a path $P_i(t)$ for each robot R_i, to minimize:

$$\sum_{i=1}^{N} \int_{t=0}^{\infty} D\left(P_i\left(t+dt\right), P_i\left(t\right)\right) dt \tag{1}$$

s.t.

$$\forall j, \exists t_j i, such\, that\, P_i\left(t_j\right) = IS_j \tag{2}$$

$$\forall tik, such\, that\, C\left(R_i, R_k\right) = 1 \tag{3}$$

$$\forall t' t'' i, such\ that \int_{t'}^{t'} D\left(P_i\left(t+dt\right), P_i\left(t\right)\right) dt \leq V \cdot (t' - t') \tag{4}$$

Here, i, $k \in \{1,2, \ldots, N\}$, $j \in \{1,2, \ldots, M\}$, t_j, t', t" $\in [0, \infty]$, t_j is the time when the first robot visits IS_j, and P$_i(t)$ is the position of $\underline{R}_i$ at time t.

Eq. (2) illustrates that each IS should be visited by some robots within a finite time. Eq. (3) demands that the robot network should keep connectivity at all times. Eq. (4) declares that robots cannot move beyond speed V. The goal is to minimize the TTD of all the robots according to Eq. (1).

4. Analysis of connectivity constraint

4.1. Optimal result of MRTPCC compared to MTSP

The MRTPCC differs from the MTSP mainly in the constraint of keeping connectivity of the robots. Intuitively, this constraint makes the TTD of solutions for the MRTPCC much longer than that of the MTSP. Theorem 1 shows that given a map of ISs and N robots, the increase in TTD caused by connectivity is bounded.

Theorem 1 *Given M ISs and N robots on a map, the communication range of robots is r. The optimal traversing path of the MRTPCC and MTSP is α and β, and the corresponding minimum TTD is $D(\alpha)$ and $D(\beta)$, so then $1 \leq \frac{D(\alpha)}{D(\beta)} \leq min(N\frac{D(\beta)+2r}{4r})$.*

Proof The minimum TTD for the MRTPCC is obviously not less than that of the MTSP, because the MRTPCC is virtually a constrained MTSP. Thus, $\frac{D(\alpha)}{D(\beta)} \geq 1$.

When $N \leq \frac{D(\beta)+2r}{4r}$, a simple solution for the MRTPCC is to make all the N robots move along βsimultaneously so that all the ISs are visited and global connectivity is maintained. For this method, the TTD is $N \bullet$D(β), and thus $\frac{D(\alpha)}{D(\beta)} \leq N$.

When $N > \frac{D(\beta)+2r}{4r}$, one can make the robots evenly distributed along β, so that the distance of adjacent robots is r, as shown in Figure 1a. The robots select a shorter way to their goal positions and travel back the same way. Thus, the TTD is $D(\alpha) \le 2 \cdot \frac{\left(r+\frac{D(\beta)}{2}\right)\cdot\frac{D(\beta)}{2r}}{2} = \frac{D(\beta)\cdot(2r+D(\beta))}{4r}$. There is $\frac{D(\alpha)}{D(\beta)} \le \frac{D(\beta)+2r}{4r}$. In this way, all the ISs are traversed. Because at least one robot is at the SS, connectivity is also maintained.

In conclusion, $1 \le \frac{D(\alpha)}{D(\beta)} \le min(N\frac{D(\beta)+2r}{4r})$ □

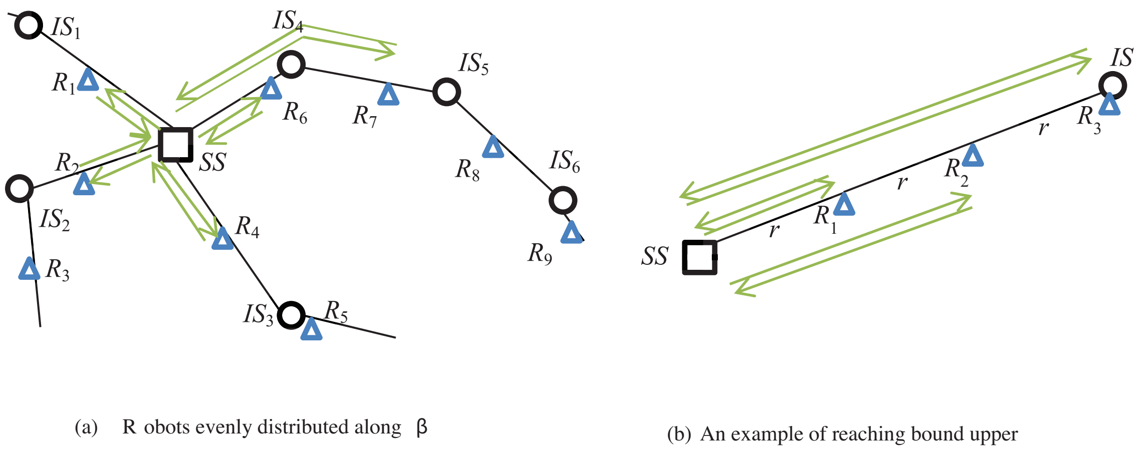

Figure 1. A solution for MRTPCC based on the optimal solution of MTSP: a) robots evenly distributed along β, b) an example of reaching the upper bound.

Note that the upper and lower bounds of $\frac{D(\alpha)}{D(\beta)}$ can be reached separately.

Reaching the lower bound: Suppose that the distances from the SS to any IS are less than r. A schedule for the MRTPCC is that one robot travels along β, and all the others stay at the SS. Therefore, there is D(α) = D(β).

Reaching the upper bound: If there are 3 robots and only one IS, which is $3r$ away from the SS, the optimal path α for the MRTPCC is shown in Figure 1b. The TTD is $12r$, while the minimum TTD for MTSP is $6r$. Hence, $\frac{D(\alpha)}{D(\beta)} = \frac{12r}{6r} = 2 = \min\left(N, \frac{D(\beta)+2r}{4r}\right) = \min\left(3, \frac{6r+2r}{4r}\right) = 2$. Theorem 1 explains the phenomenon in [28], where the increase in TTD of CNN over NN is softened as the robot number increases (where NN is a path-planning algorithm very similar to CNN, except that it does not consider the connectivity constraint). Because there is an upper bound of the ratio of the TTD of CNN to that of NN, the proof is analogous to that of Theorem 1.

5. The connection condition

In order to solve the MRTPCC, a certain condition should be followed in planning the paths. This condition, denoted as the connection condition, should ensure that the robot network is connected all the time.

Lemma 1 *Let A, B, C, D be four points on a plane, such that* $D(A,B) \le r$ *and* $D(C,D) \le r$. *For* $\forall E$ *on segment AC and* $\forall F$ *on segment BD, if* $\frac{D(A,E)}{D(A,B)} = \frac{D(B,F)}{D(B,D)}$, *then* $D(E,F) \le r$.

Proof Given that O is a reference point on the plane, and letting $\lambda=\frac{D(A,E)}{D(A,B)}=\frac{D(B,F)}{D(B,D)}$, then

$$\begin{aligned}\vec{EF} &= \vec{OF}-\vec{OE}\\ &= \left[\lambda\vec{OC}+(1-\lambda)\vec{OD}\right]-\left[\lambda\vec{OA}+(1-\lambda)\vec{OB}\right]\\ &= \lambda\vec{AC}+(1-\lambda)\vec{BD}.\end{aligned}$$

Without loss of generality, let $\left|\vec{AC}\right|\geq\left|\vec{BD}\right|$, and then $\left|\vec{EF}\right|\leq\left|\lambda\vec{AC}+(1-\lambda)\,\vec{AC}\right|\leq r$ □

Lemma 1 demonstrates that for any two robots, if the distances of their initial and goal positions are less than R, and if they move straight forward at the speed proportional to their moving distance, they will remain connected during the whole process. For instance, in Figure 2, if robot R_i moves from A to B at the speed of 3 m/s, and robot R_j moves from C to D at the speed of 4 m/s, they keep connected while they are moving.

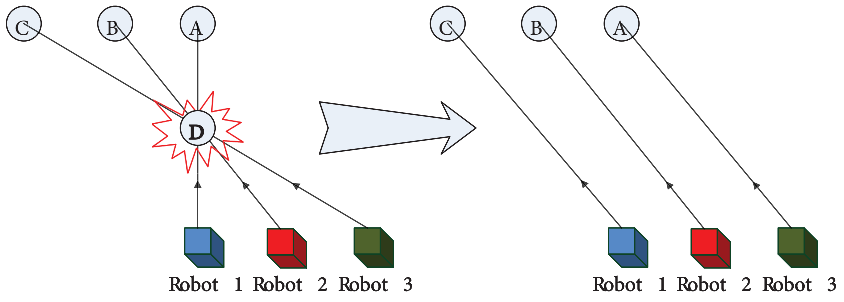

Figure 2. Connectivity maintenance of R_i and R_j; r = 30 m and V = 4 m/s.

Corollary 1 *Let A, B, C be three points on the plane, such that $D(A,B)\leq r$ and $D(A,C)\leq r$. Then for $\forall E$ on segment BC, $D(A,E)\leq r$.*

Corollary 1 is a direct use of Lemma 1. It shows a scenario in which one robot is at a standstill.

Corollary 2 *Let A_1B_1, A_2B_2, ..., A_nB_n be n segments on the plane, such that $D(A_1,A_2)\leq r, D(A_2,A_3)\leq r$, ..., $D(A_{n-1},A_n)\leq r$ and $D(B_1,B_2)\leq r, D(B_2,B_3)\leq r$, ..., $D(B_{n-1},B_n)\leq r$. For $\forall E_i$ on A_iB_i and $\forall E_{i+1}$ on $A_{i+1}B_{i+1}$, if $\frac{D(A_i,E_i)}{D(A_i,B_i)}=\frac{D(A_{i+1},E_{i+1})}{D(A_{i+1},B_{i+1})}$, then for $\forall E_j, E_k$ on A_jB_j, A_kB_k and $\frac{D(A_j,E_j)}{D(A_j,B_j)}=\frac{D(A_k,E_k)}{D(A_k,B_k)}$, such that $C(E_j,E_k)$ =1, where i, j, k $\in\{1,2,\ldots,n\}$.*

Corollary 2 is obvious according to Lemma 1. It reveals that if N robots are connected at the beginning as well as in the end, and they move straight forward at the speed proportional to their moving distance, they remain connected during the whole process.

The connection condition Let $R_1, R_2, \ldots, R_N$ be N robots; R_i is located in A_i and heads for B_i. The moving plan of these robots satisfies the connection condition if 1) $\mathrm{D}(A_1,A_2)\leq r, \mathrm{D}(A_2,A_3)\leq r, \ldots, \mathrm{D}(A_{N-1},A_N)\leq r$ and $\mathrm{D}(B_1,B_2)\leq r, \mathrm{D}(B_2,B_3)\leq r, \ldots, \mathrm{D}(B_{N-1},B_N)\leq r$; 2) R_i moves from A_i *to* B_i in a straight line; and 3) R_i moves at the speed of $\alpha\cdot\mathrm{D}(A_i,B_i)$, where $i\in\{1,2,\ldots,N\}$, $\alpha>0$.

The correctness of the connection condition is readily proven according to Corollary 2. Note that it is a sufficient condition for a path plan to be a valid solution of the MRTPCC if it satisfies the connection condition for each move. According to the connection condition, the problem can be transferred to a node division problem: divide the ISs into several sets with equal size N. In each set, the ISs are connected. Different sets may have intersections, and the union of the sets includes all the ISs. The target is to minimize the total distance of paths connecting these sets. In the next section, two algorithms are proposed to assign such paths for the robot network. By applying the connection condition, it is ensured that the algorithms produce a valid solution for the MRTPCC.

6. The solution

6.1. Connected nearest neighbor algorithm

In this subsection, the CNN algorithm is proposed. CNN works in rounds to find the next goal position and corresponding arrival time for each robot until all ISs are visited. In each round, a search phase is first performed to find a pioneer robot and IS pair based on the nearest neighbor strategy, which means the robot will move to the IS in this round. Then the restore phase is performed to find proper robots and assign their goal positions to gain connection to the pioneer robot. The robots whose goal positions are connected with that of the pioneer robot form a set S_{VR}. The restore phase repeats until all the robots are added to S_{VR}, i.e. global connectivity is achieved in this round.

Algorithm 1 CNN: A multirobot travel-planning algorithm using the nearest neighbor method while preserving connectivity.

```
Require:
  $\{IS\}$ , $\{R\}$ , $SS$, $V$ ,$r$;
Ensure:
  $Path[N][M]$, $Time[M]$;
01. $Path[1{:}N][0] = SS$, $S_{VI} = \Phi$, $S_{II} \leftarrow \{IS\}$ , $rd = 1$;
02. while $S_{II} \neq \Phi$ do
03.   $S_{VR} = \Phi$, $S_{IR} \leftarrow \{R\}$ ;
04.   for all $R_i \in S_{IR}$, $IS_j \in S_{II}$ do
05.     Find $R_x$ and $IS_y$ that minimize $D(R_i,IS_j)$;
06.   end for
07.   $Path[x][rd] = IS_y$, $R_x = IS_y$;
08.   $S_{VR} \leftarrow R_x$, $S_{VI} \leftarrow IS_y$, $S_{IR} \rightarrow R_x$, $S_{II} \rightarrow IS_y$;
09.   while $S_{IR} \neq \Phi$ do
10.     for all $R_i \in S_{IR}$, $R_k \in S_{VR}$ do
11.       if $R_i$ is connected with $R_k$ then
12.         $Path[i][rd]$=$Path[i][rd$-1];
13.         $S_{VR} \leftarrow R_i$, $S_{IR} \rightarrow R_i$;
14.       end if
15.     end for
16.     $S_{tmp} = \Phi$;
17.     for all $IS_j \in S_{II}$ do
18.       if $\exists R_i \in S_{IR}$ such that $D(R_i,IS_j) \leq r$ then
19.         $S_{tmp} \leftarrow IS_j$;
```

```
20.       **end if**
21.     **end for**
22.       **if** $S_{tmp} \neq \Phi$ **then**
23.         **for all** $R_i \in S_{VR}$, $IS_j \in S_{II}$ **do**
24.         Find $R_x$ and $IS_y$ that minimize D($R_i$,$IS_j$);
25.       **end for**
26.       $Path[x][rd] = IS_y$, $R_x = IS_y$;
27.       $S_{VR} \leftarrow R_x$, $S_{VI} \leftarrow IS_y$, $S_{IR} \rightarrow R_x$, $S_{II} \rightarrow IS_y$;
28.     **else**
29.       **for all** $R_i \in S_{VR}$, $R_k \in S_{IR}$ **do**
30.         Find $R_x$ and $R_y$ that minimize $D(R_i, R_k)$;
31.       **end for**
32.       Find $AS$ between $R_x$ and $R_y$ such that $D(AS, R_y) = r$;
33.       $Path[x][rd] = IS_y$, $R_x = AS$;
34.       $S_{VR} \leftarrow R_x$, $S_{IR} \rightarrow R_x$;
35.     **end if**
36.   **end while**
37.   **for all** $R_i$, $R_j$ whose path intersects **do**
38.     $Path[i][rd] \leftrightarrow Path[j][rd]$;
39.   **end for**
40.   $d_{\max}$ = maximum length of all paths in this round;
41.   $Time[rd] = d_{max} / V$;
42.   $rd{+}{+}$;
43. **end while**
```

The details of CNN are shown in Algorithm 1. In the search phase, the algorithm finds the shortest route from the robots to the remaining ISs and makes a virtual move (lines 4–8). The virtual move means that the robot does not physically move until the end of this round. After this step, the robot network may be disconnected, so a restore phase is followed (lines 9–36). In this phase, robots that have actually moved form a set S_{VR} (lines 10–15), and these robots are viewed to be located at their target positions. All other robots that are connected with S_{VR} are also added in, and they will stand still in this round (lines 16–27). If there are still robots outside S_{VR}, the algorithm selects the one that is nearest to S_{VR} and finds its target position to make it connected with S_{VR} (lines 29–34). Then it adds this robot into S_{VR} and repeats the previous process until all the robots are in S_{VR}. In the case that robots crash into each other, the goal positions of the cross paths are exchanged (lines 37–39), as Figure 3 shows. Note that if the cost of paths satisfies triangle inequality, the exchange will decrease the total cost. Given that the longest moving distance of the robots is $d_{\max}$ in this round, the arriving time of all robots is set to $T = d_{max} / V$, which enables them to actually move. This means that for any robot i with moving distance d_i, it will move at the speed of $V_i = d_i / T$.

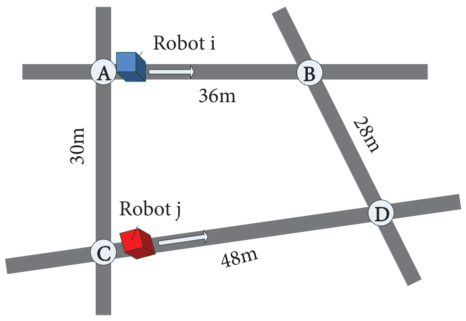

Figure 3. Collision avoidance of R_1, R_2, and R_3.

The following proposition demonstrates the accuracy of CNN:

Proposition 1 *CNN provides a valid solution for the MRTPCC, in which robots move C rounds, and in round C, robot R_i moves from Path[i][c] to Path[i][c+1] at the speed of:*

$$V_{ic} = \frac{D\left(Path\left[i\right]\left[c\right], Path\left[i\right]\left[c+1\right]\right)}{T\left[c\right]}$$

Proof By induction.

At the beginning of the CNN, the robot system is connected by definition, and at the end of the first round, the robot system is also connected according to the algorithm description. Since the robots move in straight lines at a speed proportional to their moving distance, the robot system is connected during the first round according to the connection condition.

Suppose that at the beginning of round i, the robot system is connected. It will also be connected in the end. Since robots move in straight lines at a speed proportional to their moving distance, the robot system is connected during round i.

In summary, the robot network stays connected during all C rounds.

Furthermore, CNN ensures termination due to the fact that at least one IS is visited in each round. □

Demonstration 1 *There are 3 robots and 15 ISs. The positions of the ISs and the SS are shown in Figure 4, and the communication range of the robots is set to 15. The robots start from the SS (50,50), and traverse all of the 15 ISs before they return to the SS. After applying the CNN algorithm, the resulting path for each robot is as depicted in Figure 5. In the first round, with all of the robots located at the SS, R_1 chooses IS_1 as its goal position because it is the nearest IS to the robot network. Since IS_2 is the only IS within communication range of IS_1, it is chosen by R_2 as its goal position. R_3 selects IS_3 as its target because it is the nearest to IS_1 and IS_2. During this time, all the robots are connected. They move at a speed proportional to their moving distance, and round 1 ends. In round 2, R_2 moves to IS_4, R_3 moves to IS_5, and R_1 moves to IS_6. Note that the nearest neighbor of IS_4 is R_3; however, the goal positions of R_3 and R_2 are exchanged in case of collision. In round 3, R_3 moves to IS_7, R_1 moves to IS_8, and R_2 moves to IS_9. In round 4, R_2 moves to IS_{10}, and R_1 and R_3 move to AS_1 and AS_2 respectively, to keep the connectivity of the robots. In round 4, R_2 moves to IS_{11}, R_1 moves to IS_{12}, and R_3 moves to IS_{13}. In round 5, R_3 moves to IS_{14}, R_1 moves to IS_{15}, and R_2 moves to AS_3 to maintain connectivity. By now, all the ISs are visited, and the robots return to the SS. The TTD is 618.35.*

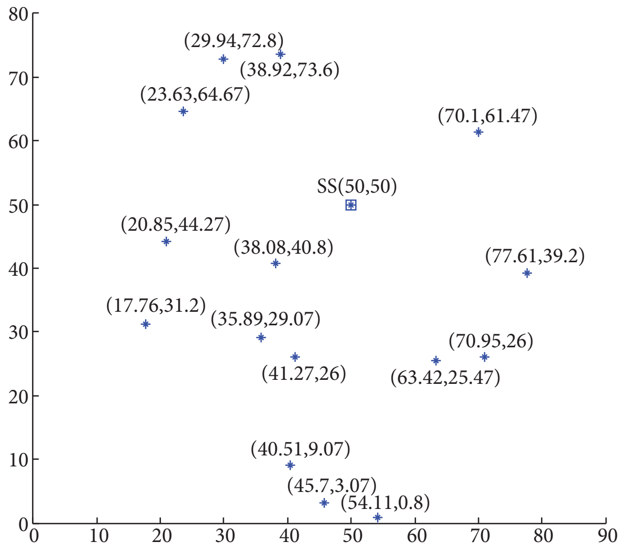

Figure 4. Positions of ISs and the SS.

6.2. B-LKH algorithm

The CNN algorithm is easy to understand and implement. However, it performs poorly because only local optimal decisions are made. It can be improved to reduce the TTD at the cost of higher computational complexity. In this section, the B-LKH algorithm is presented. The main idea of this algorithm is to globally optimize the paths of the robot network. Instead of planning paths for each robot independently, it is more reasonable to first plan the bold path for the robot network. To this end, B-LKH takes the robot network as a "large robot" with radius $r_b = \eta \cdot N \cdot r$, where $0 < \eta < 1$. When it travels on the map, it visits all the ISs covered by it. Thus, the most important issue is how to determine the travel path of the "large robot" to make the bold path covering all the ISs. B-LKH first selects a subset of ISs to cover all the ISs with circles of radius r_b. It then applies the LKH to solve the TSP of these covered centers. The resulting path is the general travel direction of the robot network. In order to traverse the ISs, B-LKH then sorts them in the order of a horizontal projection along the path. Finally, the robots visit the sorted ISs sequentially. The detail of the algorithm is shown in Algorithm 2.

Algorithm 2 B-LKH: A multirobot travel-planning algorithm based on bold Hamilton circle of covering centers.

```
Require:
  $\{IS\}$ , $\{R\}$ , $SS$, $V$ , $r$;
Ensure:
  $Path[N][M]$, $Time[M]$;
01. $Path[1{:}N][0] = SS, V_s \leftarrow \{IS\}$ , $V_c = \Phi$;
02. Calculate the distance between each $IS$; % Step 1. Bold TSP
03. for all $IS_i \in \{IS\}$ do
04.     Save its neighbors in $N_{IS}[i]$, count its neighbor number in $C_{IS}[i]$;
05. end for
```

```
06. Sort $V_s$ in descending order of $C_{IS}$;
07. for each $IS_i \in V_s$ do
08.   $V_c \leftarrow IS_i$;
09.   $V_s \rightarrow IS_i$, $V_s \rightarrow N_{IS}[i]$;
10.   Update $N_{IS}$ and $C_{IS}$;
11.   Re-Sort $V_s$;
12. end for
13. $V_c = \text{LKH}(V_c)$;
14. Adjust $V_c$ to make sure that $SS$ be the first and last element;
15. for $i = 1$ to $|V_c|$ do % Step 2. Sort the ISs
16.   $vec_1 =< V_c[i\text{-}1], V_c[i] >$;
17.   $\{IS\} \rightarrow V_c[i]$;
18.   for all $IS_i$ in $\{IS\}$ do
19.     $vec_2 =< V_c[i\text{-}1], IS_i >$;
20.     if vertical projection of $vec_2$ to $vec_1 < r$ && horizontal projection of $vec_2$ to $vec_1 \in$ [0,D($V_c[i$-
1]$,V_c[i]$)+r] do
21.       $V_t \leftarrow IS_j$;
22.       $\{IS\} \rightarrow IS_j$;
23.     end if
24.   end for
25.   Sort $V_t$ in ascending order of horizontal projection of $vec_2$ to $vec_1$;
26.   $V_s \leftarrow V_t$;
27. end for
28. $rd = 1$;
29. for $i = 1$ to $M$ do % Step 3. Traversing the ISs in order of $V_s$
30.   for $j = 1$ to $N$ do
31.     Construct minimum cost spanning tree $tr_j$ with nodes $V_s[i]$ to $V_s[i+j]$;
32.   end for
33.   Find the maximum $j$ so that the cost of $tr_j \leq N$;
34.   for $k = 0$ to $j$ do
35.     $Path[k][rd] \leftarrow V_s[i+k]$;
36.   end for
37.   for each edge $e_i$ in $tr_j$ do
38.     if $|e_i| > r$ do
39.       Put connecting points along $e_i$ into $Path[*][rd]$;
40.     end if
41.   end for
42.   for all $R_i$, $R_j$ whose path intersects do
43.     $Path[i][rd] \leftrightarrow Path[j][rd]$;
```

```
44.     end for
45.     $d_{\max}$ = maximum length of all paths in this round;
46.     $T[rd] = d_{\max} / V$;
47.     $rd$++;
48. end for
```

As shown in Algorithm 2, B-LKH works in three steps as follows: 1) to solve the bold TSP; 2) to sort the ISs; and 3) to traverse sequentially.

Solve the bold TSP (lines 2–14): First, enclose all the ISs with circles of radius r_b centered at certain ISs (lines 2–12). To select proper ISs as the covering centers, the algorithm calculates the distance between any pair of ISs. If the distance from R_i to R_j is no more than r_b, they are called neighbors. Based on the distance metrics, the neighbor list and neighbor number can be saved (lines 2–5). Then greedily select the IS with the most neighbors as one of the covering centers, and delete all the ISs covered by the circle it is centered in. After that, update the neighbor list of the rest of the ISs. These procedures are repeated until all the ISs are covered by circles (lines 6–12). Finally, the algorithm generates a bold Hamilton circle (BHC) by applying the LKH on these covering centers (lines 13–14), which shows the general travel direction of the robots.

Sort the ISs (lines 15–27): For each segment l_s of the BHC, assume its start and end points are IS_i and IS_j. B-LKH constructs a vector $vec_1 = IS_i\vec{I}S_j$, and $vec_2 = IS_i\vec{I}S_k$ for any other IS_k. Calculate the vertical and horizontal projection of vec2 on vec1. The vertical projection means the vertical distance d_v of IS_k from l_s, and the horizontal projection means a horizontal distance d_h of IS_k from IS_i. If $d_v < r_b$ and $d_h < |l_s|$+r, then IS_k will be visited by the "large robot" while it is traveling along l_s (lines 16–24). Find all the ISs satisfying the above condition, and sort them by d_h (lines 25–26). The resulting sequence is the order in which the robot network will visit.

Traverse sequentially (lines 28–48): In this step, the ISs are also traversed in rounds. In each round, the robots try to visit as many ISs as possible. To achieve this goal, the algorithm constructs a minimum cost spanning tree for the next j ISs in the visit sequence (lines 30–32). The cost means the robots needed to cover the tree. For each edge e_i, if $|e_i|$ is less than r_b, then it only needs two robots to be covered, and if $|e_i|$ is larger than r_b, then it needs at least $\lceil \frac{|e_i|}{r_b} \rceil - 1$ robots to be covered. Note that the nodes of the tree must be visited. Find the maximum j so that the corresponding spanning tree can be covered by, at most, N robots, and set the goals of these robots to the points that can cover the tree (lines 33–41). If the needed points are less than N, the residual robots try to reach the next IS in the visit sequence. After setting goal positions for all the robots in this round, the intersection paths are adjusted in the same way as Algorithm 1.

Since B-LKH satisfies the connection condition, it is certain to generate a valid solution for the MRTPCC. The proof is similar to Proposition 1. In each round, at least one IS is visited. At the end of each round, the robots stop along the generated spanning tree and are thus connected. Therefore, the robot network maintains connectivity all the time.

Demonstration 2 *Consider the example shown in Figure 3 with η = 1/3. There is r_b = 15. First, based on greedy criteria, the covering centers are selected as IS_3, IS_4, IS_7, IS_{10}, IS_{13}, and IS_{15} (SS is the starting and ending points of the paths). Apply LKH to get their Hamilton circle as shown in Figure 6a. According to the projections along the H-circle, the ISs are sorted, which is denoted by their indexes. The robots sequentially traverse these ISs as depicted in Figure 6b. In round 1, the maximum size of the minimum cost spanning tree* *(denoted by $s_{\max}$) is 3, which means the robots visit 3 ISs in this round. R_1 moves to IS_1, R_2 moves to IS_2, and R_3 moves to IS_3. In round 2, $s_{\max} = 2$. R_3 moves to IS_4, R_2 moves to IS_5, and R_1 moves to AS_1, which is near IS_6. In round 3, $s_{\max} = 3$. R_1 moves to IS_6, R_2 moves to IS_7, and R_3 moves to IS_8. In round 4, $s_{\max} = 3$. R_2 moves to IS_9, R_1 moves to IS_{10}, and R_3 moves to IS_{11}. In round 5, $s_{\max} = 3$. R_2 moves to IS_{12}, R_3 moves to IS_{13}, and R_1 moves to IS_{14}. In round 6, $s_{\max} = 1$. R_1 moves to IS_{15}, R_3 moves to AS_2 because it is near SS, and R_2 moves to SS. By now, all the ISs are visited, and R_1 and R_3 move back to SS. The TTD is 549.29.*

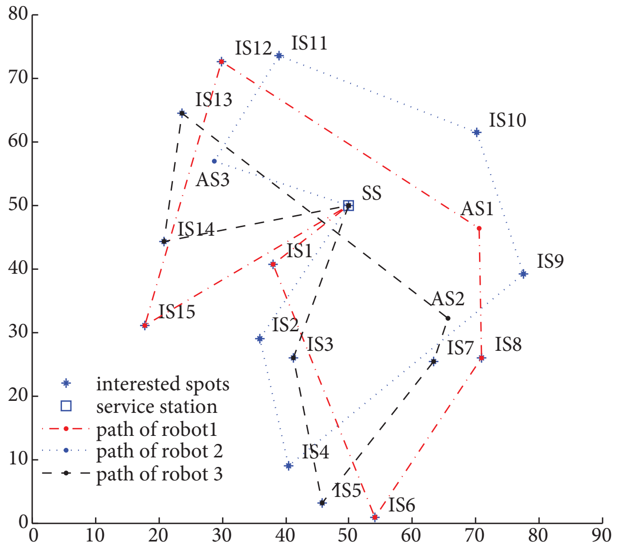

Figure 5. Travel paths of robots applying CNN.

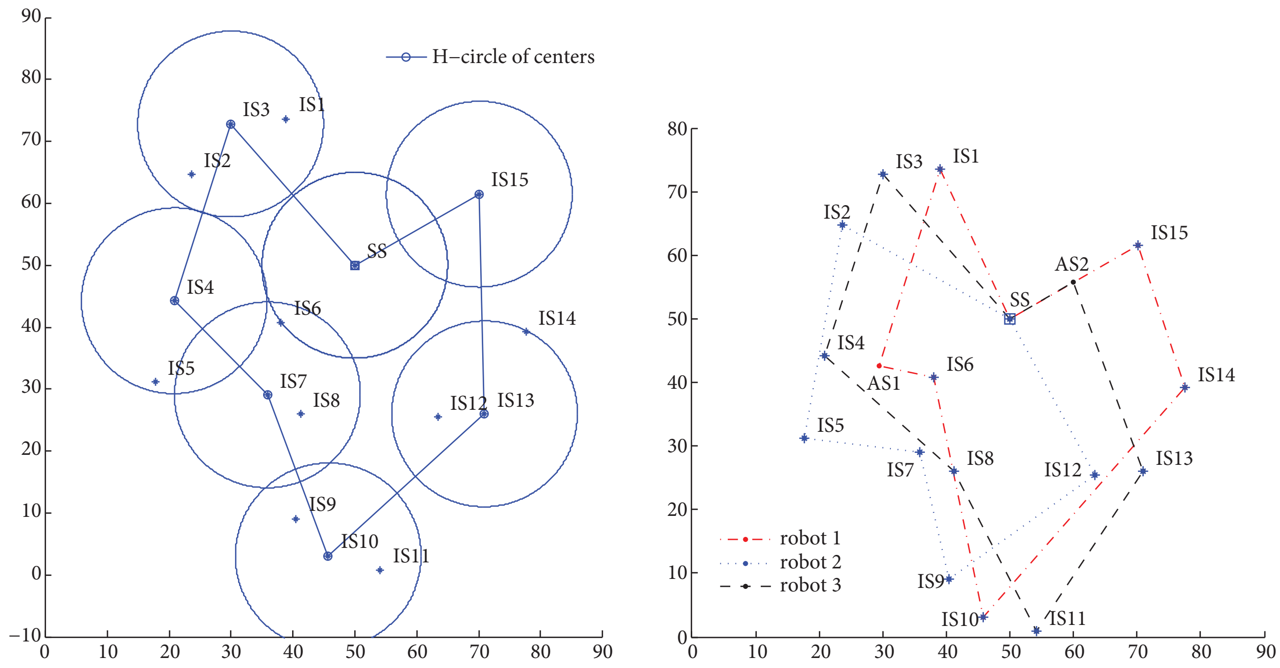

Figure 6. Travel paths of robots applying B-LKH: a) enclosing ISs, generating BHC, and sorting ISs; b) traversing paths of the robots.

6.3. Further discussion

The computational complexity of CNN is O(M^2N^3). Since $M > N$, the computational complexity of B-LKH is O(M^5), which is the same as LKH.

CNN can be implemented in either a centralized or decentralized manner. In the former case, a leader robot is elected to run the program and broadcasts goal positions for each robot. Other robots memorize the broadcasted information and move to their own goals. Heartbeat messages are regularly exchanged between the leader and the rest of the robots to monitor possible failure of the robots. In the latter case, each robot executes the same copy of the CNN to find the most beneficial strategy. A voting procedure is then performed to find a global best solution. Then the robots agree on the solution and move to their goals. Heartbeat messages are also used to keep the latest information of the environment. Furthermore, CNN is stateless. If some ISs are dynamically added in or wiped out, the robots will recalculate the next goal position according to the updated map. Since CNN ensures that the robots are connected at all times, the changing environment is regarded as a new initial state, and thus correctness is preserved.

The B-LKH is a centralized and offline algorithm. It needs to have prior knowledge of global information to plan the paths. If any IS is dynamically added in or wiped out, B-LKH has to be reexecuted to ensure correctness and performance. In addition, the performance of B-LKH is largely influenced by η, which determines the radius of the "large robot". If η is too small, the covering centers are too numerous, which increases the computation workload; additionally, the width of the path is too small, which degrades the parallel march of the robots. On the other hand, if η is too large, the robots waste energy on vertical moving, and the BHC is thus not only meaningless but also harmful. Intuitively, this is related to IS density and robot number; however, the precise analysis is left for future work. In reality, it is recommended to try different values of η and select the one with the best performance.

In general, CNN is simpler and robust to environmental dynamics. The B-LKH algorithm is more time-consuming and not so flexible, but it generates shorter travel paths. The performance of CNN and B-LKH will be extensively studied in Section 6.

7. Simulations

7.1. Simulation setup

In practice, the scale of environment ($X \times Y$), the number of ISs (M) and their distribution, and the number of robots (N) as well as their communication range (r) are the key parameters that influence the final results. Here, $\frac{M}{X \times Y}$ decides the density of the ISs, and $\frac{r}{X \times Y}$ decides the relative coverage area of robots. By default, the scale of the environment is set to 1000 $\times$ 1000 m^2 hereafter. Since the absolute velocity of the robots does not affect the results, it is set to a very large value to accelerate the simulations. The SS is located at the center of the map. The simulations mainly investigate the influence of parameters M, N, and r, as well as distributions of the ISs.

The performance of CNN and B-LKH is compared to LQCN [4]. In LQCN, each robot keeps connectivity with certain neighbors. For simplicity, a tree structure is constructed through a DFS procedure, and each robot keeps connectivity with its parent robot in the tree. For comparison, the simulations assume the acceptable range of each robot as r. Since LQCN considers only one goal for each robot, we adopt the allocation method used in [14] and [15] to solve the whole traveling problem. Each robot automatically selects its goal position (an IS) to maximize the information gain of $G = \frac{1.5^x}{1.5^y \cdot d}$, where x and y are IS number and robot number within the sensing range of the IS, and d is the distance from the robot to the IS.

Note that the absolute value of TTD is not important, since it is largely influenced by the map. Therefore, all the results are ratios to the TTD of the nearest neighbor (NN) method. NN is very similar to CNN except that NN takes no consideration of connectivity. The reason to use NN as a benchmark algorithm is that its result is little influenced by N and r, which is suitable for normalization. Additionally, it also applies the nearest neighbor strategy, which minimizes its distinctions with CNN.

All the results in the simulations are averaged by 200 runs.

7.2. Simulation results and analysis

Effect of the amount of robots: In this simulation, r = 50 m, M = 100, and ISs are subject to uniform distribution. N varies from 2 to 8.

Figure 7 shows the TTD of CNN, LQCN, and B-LKH (η = 1/N, 0.25, 0.5, 0.75) over NN. In this simulation, LQCN performs worse than CNN and B-LKH. B-LKH performs better than CNN except when η = 0.75 and N $>$ 5. As discussed before, η largely influences the performance of B-LKH, so it should be neither too large nor too small. As depicted in Figure 7, B-LKH performs best when η takes values of 0.25 and 1/N. In addition, when $\eta = 0.25$, B-LKH also performs well, although not always the best, in the following simulations, so only the results of B-LKH with $\eta = 0.25$ are shown hereafter. As the number of robots increases, the TTD generated by CNN, LQCN, and B-LKH increases relative to NN. Since the TTD of NN changes little, Figure 7 reflects the trend of the absolute value of these algorithms. The reason for the increase is that more TTD is wasted to keep the robot network connected. Under the connectivity constraint, when the robot network decides to traverse some ISs, it may have to give up visiting other ISs even if they are close to some of the robots. Otherwise the robot network will be partitioned. These nearby ISs can only be visited at some point in the future, thus increasing the TTD. However, according to Theorem 1, if N is large enough, the increase of the TTD relative to NN will tend to be zero.

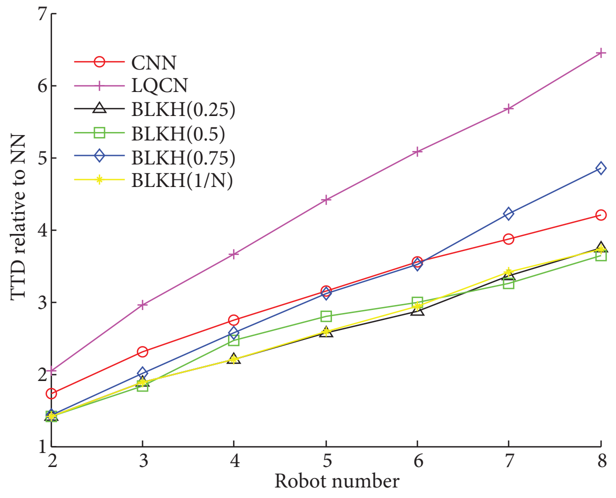

Figure 7. TTD comparison among NN, CNN, LQCN, and B-LKH varying N.

Effect of the amount of ISs: In this simulation, r = 50 m, N = 5, and ISs are subject to uniform distribution. M varies from 20 to 200.

As depicted in Figure 8, in this simulation, B-LKH still performs the best, while LQCN performs the worst. There are two main reasons for the larger TTD of LQCN. First, robots act in a distributed manner in LQCN, resulting in two or more robots possibly setting the same goal position. Only one of them will achieve the goal and let the others go to other goals. This is aggravated when IS density becomes high. Second, the selection of goals does not ensure that robots will keep connected during the whole process. Instead, achieve-connection behavior is constantly carried out, which further increases the TTD. When IS density is low, this behavior occurs frequently. As the number of ISs increases, the TTD generated by CNN and B-LKH decreases relative to NN. This is because the higher the density of ISs, the less TTD is wasted for keeping connectivity. Robots have more choices when selecting the next target if the density is sufficiently high, and thus the cost of keeping connectivity is reduced. The theoretical lower bound of the ratio is 1. Note that when M increases, the TTD of NN also increases.

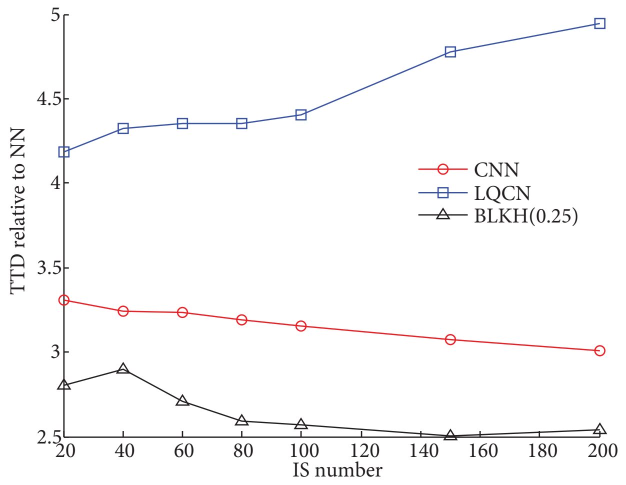

Figure 8. TTD comparison among NN, CNN, LQCN, and B-LKH varying M.

Effect of distribution of ISs: To investigate how the distribution of ISs influences the results, three other distributions are designed. They are: 1) Grid distribution, where ISs are distributed on the vertexes of grids whose side length is 100 m. This distribution corresponds to street surveillance. 2) Circular distribution, where ISs are uniformly distributed along a circle whose radius is 495 m. This distribution corresponds to boarder surveillance. 3) Normal distribution, where ISs are subject to normal distribution. This distribution corresponds to priority-based surveillance.

Suppose that $N = 5$, $M = 100$, and r ranges from 25 m to 125 m. The results are depicted in Figure 9, in which Figure 9a shows the result of uniform distribution. Figure 9b shows the result of grid distribution, Figure 9c shows the result of circular distribution, and Figure 9d shows the result of normal distribution. Generally, B-LKH performs the best, while LQCN performs the worst. When the communication range increases, the connectivity constraint is softened, which leads to a decrease in the TTD of CNN, B-LKH, and LQCN. In extreme cases, when r $\to \infty$, the TTD of CNN will be the same as NN. The simulation results validate this proposition. Compared to NN, all the algorithms generate less TTD as r increases, except for LQCN in Figure 9c. This is because in circular distribution, the density of ISs is very high along the circle. As discussed before, the target of each robot conflicts more often, and thus robots waste more distance on avoiding collision. CNN, LQCN, and B-LKH perform the poorest relative to NN in Figure 9c, which is because almost all of the robots are passively moved along the circle to keep connectivity. The most interesting part of the results is the turning points of the curves in Figure 9b. The turning points appear at $r = 100$ m, which is exactly the side length of the grids. The ratio decreases slightly when $r > 100$ m. This phenomenon may reveal the relationship between the ratio and the mean number of the ISs' neighbors that are not r far away. The results in Figures 9a and 9d are similar, in which CNN and B-LKH significantly outperform LQCN. Although the TTD of LQCN decreases in these conditions, avoiding collision and achieving connection behaviors makes the decrease relatively smoother.

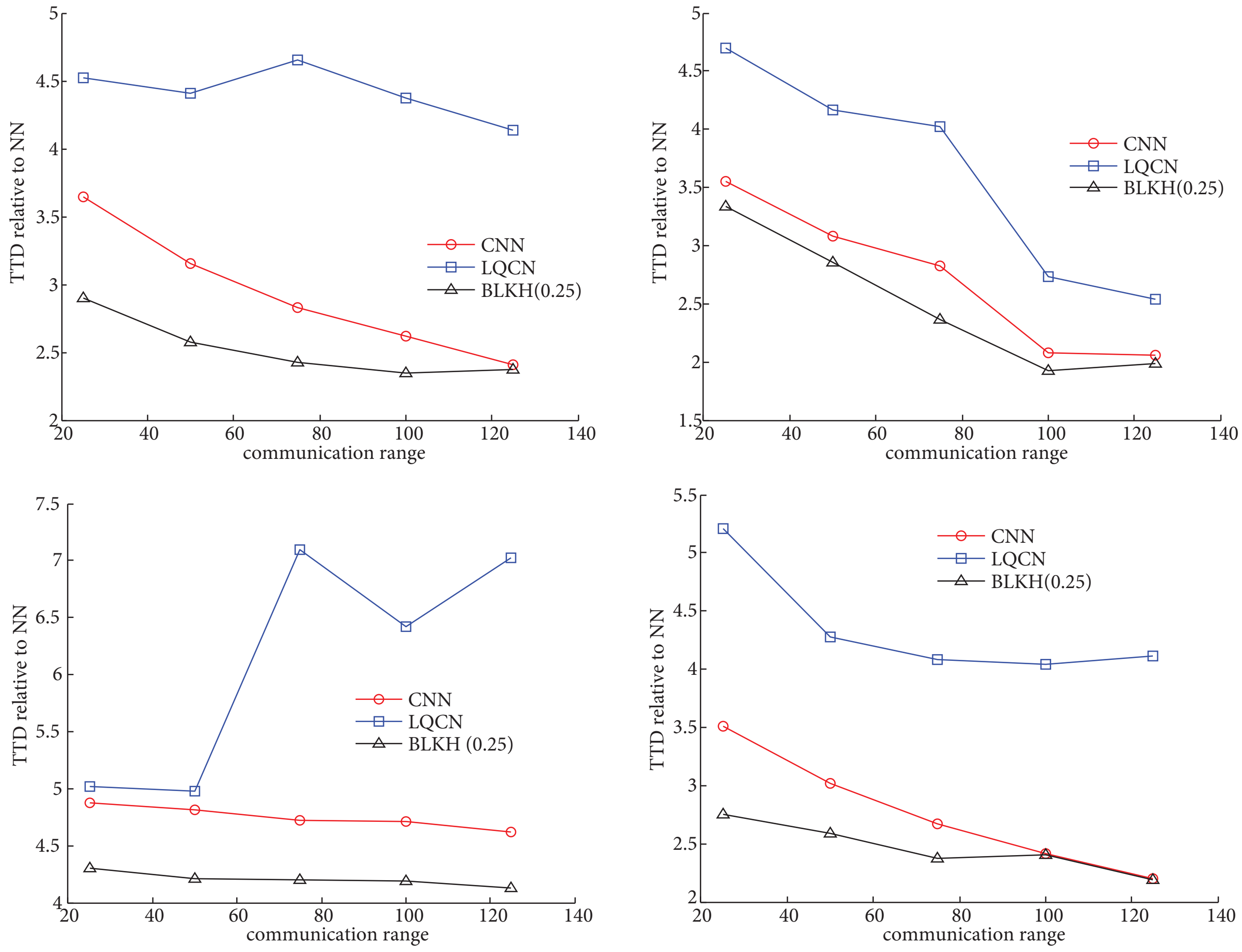

Figure 9. TTD comparison varying r under different distributions of ISs: a) uniform distribution, b) grid distribution, c) circular distribution, d) normal distribution.

Summary From these simulations, it can be seen that B-LKH and CNN significantly outperform LQCN in all conditions, because B-LKH and CNN explicitly consider the connectivity constraint in planning the paths. Compared with B-LKH and CNN, LQCN wastes too much energy on avoiding conflicts and achieving connection, and thus it consumes a much higher TTD. B-LKH (0.25) performs better than CNN because CNN makes decisions by only relying on local information, while B-LKH preprocesses global information and is able to produce better solutions. From Figure 7, it is clear that the value of ηinfluences the performance of B-LKH dramatically, and hence it should be carefully considered.

8. Conclusion

This paper formulates the MRTPCC, which requires minimizing the TTD of the robots when they are traversing ISs. The paper then analyzes the bound of increase in TTD caused by connectivity constraints and proposes and proves the connection condition, which is used to direct the design of solutions. Based on the connection condition, CNN and B-LKH are proposed to minimize the TTD, which work in rounds to ensure connectivity of the robot network. CNN is simpler and robust to environmental dynamics, while B-LKH is more time-consuming and generates shorter travel paths. Simulation results show that the proposed algorithms perform better when the number of ISs increases and worse when the number of robots increases relative to the NN method. Influences of distribution of ISs are also studied. Through comparison, it is depicted that the proposed algorithms significantly outperform LQCN in TTD.

In future work, an environment with irregularly shaped obstacles will be studied, and real-world experiments are in progress. The proper number of robots to tackle certain surveillance tasks while preserving connectivity will also be investigated. The connectivity of several local networks will be considered instead of the global connectivity of the whole network. In each local robot network, the connectivity of a certain number of robots shall be guaranteed.

Acknowledgments

First of all, we would like to thank the reviewers and the editor for the positive and constructive comments and suggestions. This research was partially supported by the Natural Science Foundation of China under Grant No. 60973122, the 973 Program in China under Grant No. 2009CB320705, and the 863 Hi-Tech Program in China under Grant No. 2011AA040502.

References

[1] Sugiyama H, Tsujioka T, Murata M. Integrated operations of multi-robot rescue system with ad hoc networking. In: IEEE Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology; 17–20 May 2009; Aalborg, Denmark: IEEE. pp. 535–539.

[2] Akat SB, Gazi V, Marques L. Asynchronous particle swarm optimization-based search with a multi-robot system: simulation and implementation on a real robotic system. Turk J Electr Eng Co 2010; 18: 749–764.

[3] Topal S, Erkmen I, Erkmen AM. A novel multirobot map fusion strategy for occupancy grid maps. Turk J Electr Eng Co 2013; 21: 107–119.

[4] Hsieh MA, Cowley A, Kumar V, Taylor CJ. Maintaining network connectivity and performance in robot teams. J Field Robot 2008; 25: 111–131.

[5] Arkin RC, Diaz J. Line-of-sight constrained exploration for reactive multiagent robotic teams. In: International Workshop on Advanced Motion Control; 2002: IEEE. pp. 455–461.

[6] Vazquez J, Malcolm C. Distributed multirobot exploration maintaining a mobile network. In: 2nd International IEEE Conference on Intelligent Systems; 22–24 June 2004: IEEE. pp. 113–118.

[7] Anderson SO, Simmons R, Golberg D. Maintaining line of sight communications networks between planetary rovers. In: 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems; 23–27 October 2003: IEEE. pp. 2266–2272.

[8] Ji M, Egerstedt M. Distributed coordination control of multiagent systems while preserving connectedness. IEEE T Robot 2007; 23: 693–703.

[9] Dimarogonas DV, Kyriakopoulos KJ. Connectedness preserving distributed swarm aggregation for multiple kinematic robots. IEEE T Robot 2008; 24: 1213–1223.

[10] Sabattini L, Gasparri A, Secchi C, Chopra N. Enhanced connectivity maintenance for multi-robot systems. Robot Contr 2012; 10: 319–324.

[11] Esin YH, Ünel M. Formation control of nonholonomic mobile robots using implicit polynomials and elliptic Fourier descriptors. Turk J Electr Eng Co 2010; 18: 765–780.

[12] Datla D, Chen X, Tsou T, Raghunandan S, Shajedul Hasan S, Reed JH, Dietrich CB, Bose T, Fette B, Kim J. Wireless distributed computing: a survey of research challenges. IEEE Comm Mag 2012; 50: 144–152.

[13] Bonnet F, Défago X. Exploration and surveillance in multi-robots networks. In: Second International Conference on International Conference on Networking and Computing; 30 November – 2 December 2011; Osaka, Japan: IEEE. pp. 342–344.

[14] Stump E, Michael N. Multi-robot persistent surveillance planning as a vehicle routing problem. In: 2011 IEEE Conference on Automation Science and Engineering; 24–27 August 2011; Trieste, Italy: IEEE. pp. 569–575.

[15] Michael N, Stump E, Mohta K. Persistent surveillance with a team of MAVs. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems; 25–30 September 2011; San Francisco, CA, USA: IEEE. pp. 2708–2714.

[16] Trevai C, Ota J, Arai T. Multiple mobile robot surveillance in unknown environments. Adv Robot 2007; 21: 729–749.

[17] Khamis AM, Elmogy AM, Karray FO. Complex task allocation in mobile surveillance systems. J Intell Robot Syst 2011; 64: 33–55.

[18] Bektas T. The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega 2006; 34: 209–219.

[19] Bellmore M, Hong S. Transformation of multisalesman problem to the standard traveling salesman problem. J ACM 1974; 21: 500–504.

[20] Rao MR. Technical note—A note on the multiple traveling salesmen problem. Oper Res 1980; 28: 628–632.

[21] Xu Z, Rodrigues B. A 3/2-approximation algorithm for multiple depot multiple traveling salesman problem. In: 12th Scandinavian Symposium and Workshops on Algorithm Theory; 21–23 June 2010; Bergen, Norway: Springer. pp. 127–138.

[22] Laporte G. A concise guide to the traveling salesman problem. J Oper Res Soc 2010; 61: 35–40.

[23] Lin S, Kernighan BW. An effective heuristic algorithm for the traveling-salesman problem. Oper Res 1973; 21: 498–516.

[24] Helsgaun K. An effective implementation of the Lin–Kernighan traveling salesman heuristic. Eur J Oper Res 2000; 126: 106–130.

[25] Derbakova A, Correll N, Rus D. Decentralized self-repair to maintain connectivity and coverage in networked multi-robot systems. In: IEEE International Conference on Robotics and Automation; 9–13 May 2011; Shanghai, China: IEEE. pp. 3863–3868.

[26] Ponda SS, Johnson LB, Kopeikin AN, Choi H, How JP. Distributed planning strategies to ensure network connectivity for dynamic heterogeneous teams. IEEE J Sel Area Comm 2012. 30: 861–869.

[27] Zhou G, He T, Krishnamurthy S, Stankovic JA. Models and solutions for radio irregularity in wireless sensor networks. ACM T Sens Netw 2006. 2: 221–262.

[28] Cheng H, Yun W, Fei B. Multi-robot traveling problem constrained by connectivity. In: ICPCA/SWS; 28–29 November 2012; İstanbul, Turkey: LNCS. pp. 231–245.