

Article

A Two-Stage Method for Target Searching in the Path Planning for Mobile Robots

Tao Song ¹, Xiang Huo ¹ and Xinkai Wu ^{1,2,*}

¹ School of Transportation Science and Engineering, Beihang University, Beijing 100193, China; songtaobuaa@buaa.edu.cn (T.S.); huoxiang@buaa.edu.cn (X.H.)

² Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100193, China

* Correspondence: xinkaiwu@buaa.edu.cn; Tel.: +86-010-8231-6713

Received: 23 October 2020; Accepted: 2 December 2020; Published: 3 December 2020



Abstract: The path planning for target searching in mobile robots is critical for many applications, such as warehouse inspection and caring and surveillance for elderly people in the family scene. To ensure visual complete coverage from the camera equipped in robots is one of the most challenging tasks. To tackle this issue, we propose a two-stage optimization model to efficiently obtain an approximate optimal solution. In this model, we first develop a method to determine the key locations for visual complete coverage of a two-dimensional grid map, which is constructed by drawing lessons from the method of corner detection in the image processing. Then, we design a planning problem for searching the shortest path that passes all key locations considering the frequency of target occurrence. The testing results show that the proposed algorithm can achieve the significantly shorter search path length and the shorter target search time than the current Rule-based Algorithm and Genetic Algorithm (GA) in various simulation cases. Furthermore, the results show that the improved optimization algorithm with the priori known frequency of occurrence of the target can further improve the searching with shorter searching time. We also set up a test in a real environment to verify the feasibility of our algorithm.

Keywords: targets search; mobile robot; path planning; visual complete coverage; two-stage

1. Introduction

In recent years, the application of intelligent mobile robots in a smart home system has attracted many attentions. In a home environment, mobile robots equipped with multiple sensors are often used to perform some searching tasks, such as searching for people, pets, or objects. Due to the complexity of the indoor environment with many obstacles, generating a search path plan for a specific target is a big challenge for robots. In particular, it becomes a difficult task to quickly generate an efficient search path plan for a mobile robot equipped with a camera. In fact, with the known map information, the search task using cameras is a vision coverage path planning (VCP) problem. This problem can be regarded as a variant problem derived from the coverage path planning (CPP) problem. CPP essentially is used to determine a path that passes all points of an area or volume of interest while avoiding obstacles [1]. CPP has been widely used in many robotic applications, such as vacuum cleaning robots, construction waste recycling robots, autonomous underwater vehicles, creating image mosaics, demining robots, lawn mowers, automated harvesters, window cleaners, and inspection of complex structures [2–6].

A considerable body of research has studied the CPP problem (e.g., [1,6–9]), but relatively little research has been conducted on VCP. Intuitively speaking, VCP is a combination of solving a Watchman Route Problem (that is, a problem of computing the shortest route from Point A to Point B in a known area) and an Art Gallery Problem (that is, a problem of selecting the minimum

number of observation points to completely observe a known area). Finding an optimal solution to this combination is obviously a daunting task. Wang et al. [10] used an approximation method to find a bounded suboptimal solution, which considers both traveling time and sensing time with respect to a pre-defined set of sensing positions. However, the problem of selecting the initial set of sensing positions remains open and the approach presents scalability issues when the cardinality of the set is increased. Some other algorithms have been proposed for the optimal solution of VCPP; but these algorithms are only effective with restricted assumptions, such as not considering occlusions in the field of view of the sensors [11], or the detection environment is relatively simple [12]. These solutions present two major drawbacks: (1) these methods are difficult to efficiently solve the problem in some complex scenarios; and (2) they do not consider the cost of moving from one search location to another. Please note, in a complicated indoor environment, the mobile robot moving cost between key search locations is critical and must be considered throughout the target search path planning process. Therefore, after determining key search locations, we will still need to solve a (Metric) Traveling Salesman Problem to find the shortest tour that connects all the search locations [13]. It is a well-known NP-hard problem, although it is possible to optimally solve very large TSP instances, with hundreds of locations [14].

This research aims to tackle the VCPP problem in a complex indoor environment for mobile robots. Our main work includes: (1) Propose an efficient method to determine the key locations for visual complete coverage of a two-dimensional grid map, which is constructed by drawing lessons from the method of corner detection in the image processing. As the field of view of robot camera can completely search the entire known area, the union of the area covered by the visual field of the robot camera at the key locations can achieve full coverage of the whole environment. (2) Design a planning problem for searching the shortest path that passes all key locations considering the frequency of target non-occurrence. The Ant Colony Optimization (ACO) algorithm will be used to solve this problem.

The rest of the article is structured as follows. We define the path planning problem of target search with a mobile robot in Section 2, followed by a two-stage optimization model in Section 3. Section 4 presents the detailed methods of our model. In Section 5, multiple simulation and experimentation results are conducted for comparisons and discussion. Finally, we conclude the article in Section 6.

2. Problem Description

The proposed VCPP solution is applied to a mobile robot with fundamental functions including motion control, sensor information fusion, simultaneous mapping, navigation capabilities, etc. Target searching is carried out by a camera mounted on a 360-degree controllable vision pan-tilt unit. Real-time images at current location will be processed by image process algorithm to extract target information. During the process, we also assume the environment is static and the locations of walls or other obstacles will not be changed.

To increase the cover range of the robot camera, a 360-degree controllable vision pan-tilt unit is equipped, and the radius of search circular is assumed r . Therefore, the robot can detect an area with 360-degree view. Note we assume the working environment for mobile robot is known; and the grid map is derived by the widely used coverage algorithm developed by Mansouri et al. [9]. In the grid map, the side length of the square grid is determined by the size of the robot outline plus a safe distance. Therefore, the target area can be defined as a Cartesian grid with a set of all identical grids, A . Within A , the subset O covers all grids containing obstacles which stop the detection of the robot camera and the movement of the robot; and the subset S includes N grids which do not contain obstacles, i.e., $S = \{1, \dots, N\}$, as shown in Figure 1. We assume that every passable grid is reachable for the robot.

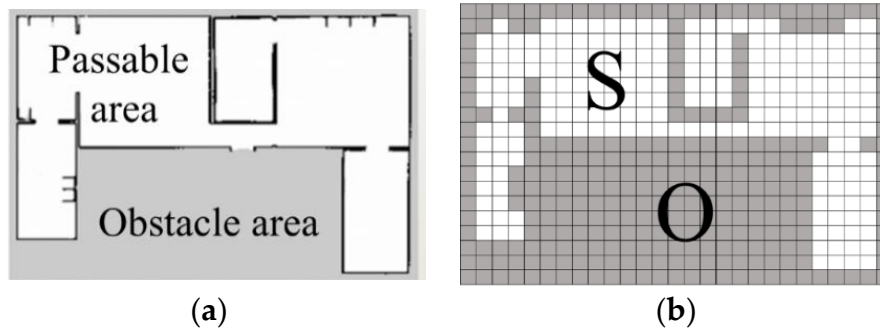


Figure 1. (a) Environment map; (b) grid map.

To complete the target searching, we need to first determine an optimal path which can be passed by the robot. We then define the optimal path as a set, P , an ordered column vector with the first cell as the starting location of the mobile robot. As P is defined as an optimal path, then the moving cost associated to P is minimal; and the visual coverage covers all passable grids through a series of observation locations of P . These observation locations in P are defined as $\{p_1, \dots, p_i, \dots, p_n\}$, i.e., $P = \{p_1, \dots, p_i, \dots, p_n\}$. In addition, $p_i \rightarrow p_{i+1}$ is the subpath of the optimal path P .

An example of these steps is detailed in Figure 2. The simple test map is shown in Figure 2a, where obstructed grids are represented in black and passable ones in white. The passable grids have been marked with number 1, \dots , 36. As shown in Figure 2b, the optimal path P ($2 \rightarrow 5 \rightarrow 24 \rightarrow 33$) is determined by a minimum length shortest path, i.e., $P = \{2, 5, 24, 33\}$.

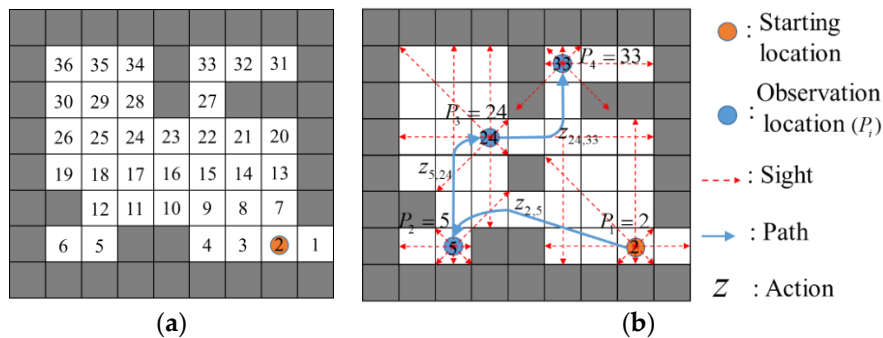


Figure 2. (a) Test map; (b) optimal path P ($2 \rightarrow 5 \rightarrow 24 \rightarrow 33$).

3. A Two-Stage Model for VCPP

Before we present the model, the list of the notations is summarized in Table 1.

3.1. Mathematical Model

We aim to find the optimal observation location configuration P with minimum moving cost by solving the following problem:

Minimize

$$\sum_{i=1}^{n-1} c(z_{p_i, p_{i+1}}) \quad (1)$$

subject to

$$\cup v(Z) = S \quad (2)$$

$$p_1 = g \quad (3)$$

$$P \subseteq S \quad (4)$$

where Equation (1) is the objective function of the model. The moving cost includes the path distance generated by the subpath of P performed by the mobile robot. $c(z_{p_i, p_{i+1}})$ is the moving cost associated with the subpath $z_{p_i, p_{i+1}}$ ($p_i \rightarrow p_{i+1}, p_i, p_{i+1} \in P$).

Table 1. Primary notation.

Parameters	
S	set of passable grids
P	optimal path
n	number of observation locations of P
i	the ID of the observation locations of $P, i \in \{1, \dots, n\}$
p_i	observation location, $p_i \in P$
Z	set of subpath of P
$z_{p_i, p_{i+1}}$	a subpath which refers that robot moves from p_i to $p_{i+1}, z_{p_i, p_{i+1}} \in Z$
$\cup v(Z)$	union of a series of visible grid sets detected by Z
$c(z_{p_i, p_{i+1}})$	the moving cost associated with $z_{p_i, p_{i+1}}, z_{p_i, p_{i+1}} \in Z$
g	the ID of the starting location of a mobile robot
Ω	set of key locations
α_k	a binary indicating if the grid k is set as a key location, $k \in S$
$v(k)$	set of passable grids that the robot can detect at key location k
$\cup v(\Omega)$	union of passable grids visible that the robot can detect at all key locations $\in \Omega$
N_Ω	number of key locations in Ω
$x_{j,k}$	binary factor, which indicates whether the mobile robot moves from key location j to $k, j, k \in \Omega$
$d(j, k)$	distance between key location j and $k, j, k \in \Omega$
f_k	probability of the robot unable to search for the target at the key location $k, k \in \Omega$
w_k	number of test times of the robot unable to search for the target at key location $k, k \in \Omega$
W	total number of test times
Φ	subset of set Ω

$\cup v(Z)$ is the union of a series of visible grid sets detected by Z . S is the set of passable grids. All grids detected by the observation locations of P include all passable grids, as shown in Equation (2).

g is the starting location of the mobile robot. The first cell of P p_1 is set as the starting location of the mobile robot, as expressed in Equation (3).

The ordered column vector $P = \{p_1, \dots, p_n\}$, which expresses the optimal path passed by the robot, is the decision variable. A series of grids $\in S$ make up the ordered column vector P . P is a subcollection of S , as shown in Equation (4).

3.2. A Two-Stage Model Design

It is difficult to find an optimal subcollection P of S to address the problem of target search in a complex grid map due to the computational complexity. To improve computational efficiency, we can obtain an approximate optimal solution by structuring a two-stage optimization model. In the first stage, we focus on minimizing observation locations to achieve vision complete coverage of all grids in S ; and in Stage II, we address the path planning problem by connecting these observation locations. The diagram of the two-stage optimization program is shown in Figure 3. In brief, in this proposed two-stage optimization program, Stage I is modeled as an observation location determination problem (OLDP); and Stage II is designed as an observation location connectivity problem (OLCP).

(1) Stage I: An OLDP model: The basic idea is to lock some observation locations by thoroughly observing all grids in S . The excellent method used by some authors to find a pre-defined set of sensing locations to completely observe a known area (e.g., [10,13]). However, these methods are poorly implementable when the cardinality of the set of sensing positions is large. We focus on minimizing observation locations to achieve vision complete coverage of all grids in S . The observation locations are defined as “key locations”. The detailed OLDP modeling is described as follows:

Minimize

$$\sum_{k \in S} \alpha_k \quad (5)$$

subject to

$$\alpha_k = \begin{cases} 1 & \text{if } k \in \Omega \\ 0 & \text{otherwise} \end{cases} \quad k \in S \quad (6)$$

$$\cup v(\Omega) = S \quad \Omega \subseteq S \quad (7)$$

$$\alpha_g = 1 \quad (8)$$

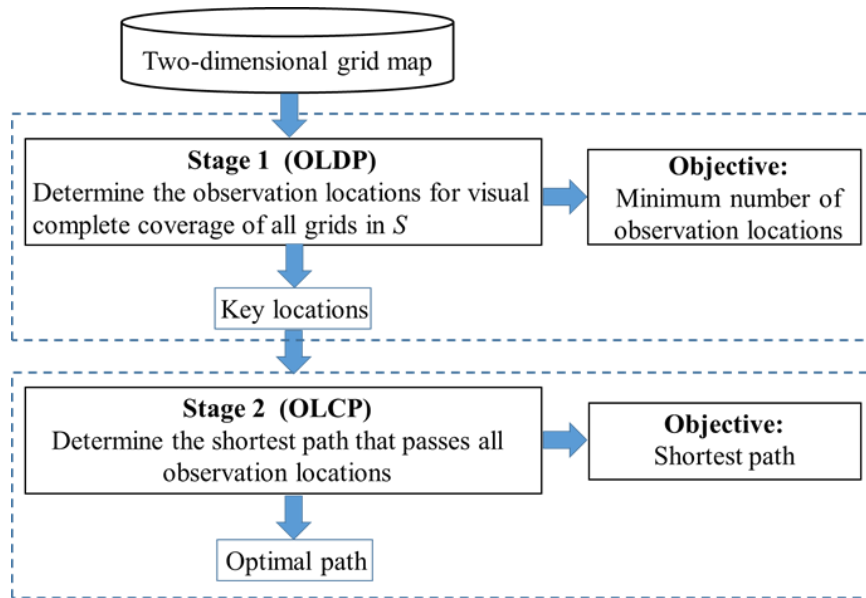


Figure 3. The diagram of the two-stage optimization program.

Objective function: Equation (5) is the objective function of the OLDLP model. As few observation locations as possible are obtained to solve VCPP.

Constraints: After we construct the set of key locations as Ω , we introduce a binary variable α_k , where $\alpha_k = 1$ when the grid k is set as a key location and $\alpha_k = 0$ otherwise, as shown in Equation (6).

In Equation (7), $\cup v(\Omega)$ denotes the union of all visible grid sets detected by the robot passing all key locations of Ω . Equation (7) explains that the area seen by the robot passing all key locations will cover all grids in S .

Equation (8) explains the starting location g of the mobile robot belongs to Ω .

A heuristic method is used to solve OLDLP, as will be introduced in Section 4.1.

(2) Stage II: An OLCPP model: After the set of key locations Ω is obtained, we then search the shortest path which connects these key locations, which is modeled as an OLCPP problem.

OLCPP is similar to the traveling salesman problem (TSP). However, the classic TSP requires that the traveler departs from the starting point and returns to the starting point after passing through all locations. In our model, we search for a shortest path which visits each key location in Ω but without going back to the starting location. Therefore, our planned path will not form a closed loop, i.e., Hamiltonian cycle. To still apply the TSP model, we add a virtual location to Ω , set the distance from this virtual point to the starting location is 0 and set the distance from this virtual point to other key locations is the maximum distance between key locations. After adding a virtual location, we solve the TSP problem to plan a looped path and then remove the virtual points to get the optimized path.

In addition, a critical point for our research is that the shortest path that passes all key locations is determined considering the frequency of target non-occurrence of each location. The more difficult it is to observe the target in some locations, the less it should be prioritized to go to these locations for robot when designing the path. In other words, the higher the moving cost to reach these locations should be.

To formulate the above-mentioned frequency problem, we introduce a parameter f_k , which indicates the frequency of the robot which is unable to search for the target at the key location k . When the robot moves to location k without finding the target, the path distance to location k is the real moving cost for our model. The moving cost from any key location j to key locations k is estimated as the product of $d(j, k)$ and f_k . $d(j, k)$ is the path distance between key locations j and k .

Then, this OLCP model with frequency can be described in the following:

Minimize

$$\sum_{j \in \Omega} \sum_{k \in \Omega} (x_{j,k} \cdot d(j, k) \cdot f_k) \quad j \neq k \quad (9)$$

subject to

$$x_{j,k} = \begin{cases} 1 & j \text{ and } k \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \quad j, k \in \Omega, \quad j \neq k \quad (10)$$

$$f_k = \frac{\frac{w_k}{W}}{\sum_{k \in \Omega} (\frac{w_k}{W})} \quad (11)$$

$$\sum_{j \in \Omega} x_{j,k} = 1 \quad j, k \in \Omega, \quad j \neq k \quad (12)$$

$$\sum_{k \in \Omega} x_{j,k} = 1 \quad j, k \in \Omega, \quad j \neq k \quad (13)$$

$$\sum_{j \in \Omega} \sum_{k \in \Omega} x_{j,k} \leq |\Phi| - 1 \quad j, k \in \Phi, \quad \forall \Phi \subset \Omega, \quad 2 \leq |\Phi| \leq N_\Omega \quad (14)$$

In detail, the OLCP model consists of:

Objective function: Equation (9) is the objective function of the OLCR problem. It searches the optimal path with shortest distance which connects all key locations in Ω . The binary factor $x_{j,k}$ indicates whether the mobile robot moves from key location j to k . $d(j, k)$ illustrates the distance between key locations j and k . f_k illustrates the frequency of the robot which is unable to search for the target at the key location k .

Constraints: $x_{j,k}$ indicates whether the mobile robot moves from key location j to k is the decision variable in this model, as formulated by Equation (10).

w_k is the number of test times of the robot unable to search for the target at key location k . W is the total number of target search test times. The frequency that the robot unable to search for the target at the key location k is $\frac{w_k}{W}$. It means that the mobile robot passes the key location k w_k times but cannot find the target in total W times tests. After normalization, this frequency value f_k is $\frac{\frac{w_k}{W}}{\sum_{k \in \Omega} (\frac{w_k}{W})}$, as shown in Equation (11). In the absence of multiple test data for searching the target in the working environment (i.e., no prior knowledge of target non-appearance frequency), f_k is set to 1.

Equations (12) and (13) explain that for each key location, only one path enters the key location and only one path moves out of the key location. Equation (14) guarantees that the model does not have any sub-circuit solutions. In Equation (14), Φ is a subset of set Ω . The number of key locations in Φ is within 2 and N_Ω .

The Ant Colony Optimization algorithm (ACO) is used to solve OLCP, as will be introduced in Section 4.2.

4. Solving Methods

4.1. Stage I: A Heuristic Method

It is challenging to find the key locations in a complex grid map. Through our observations, we find that searching key locations using the boundary grids will significantly reduce the complexity of the algorithm. Therefore, we propose the following two-phase heuristic method: first, we use the pixel-level corner detection method in the image processing to find the boundary grids in the “relative macro” grid map scene used by the mobile robot; and second, we search the key locations in these boundary grids, instead of a complex analytical solution process. The detailed procedure is described in the following steps and the flowchart in Figure 4.

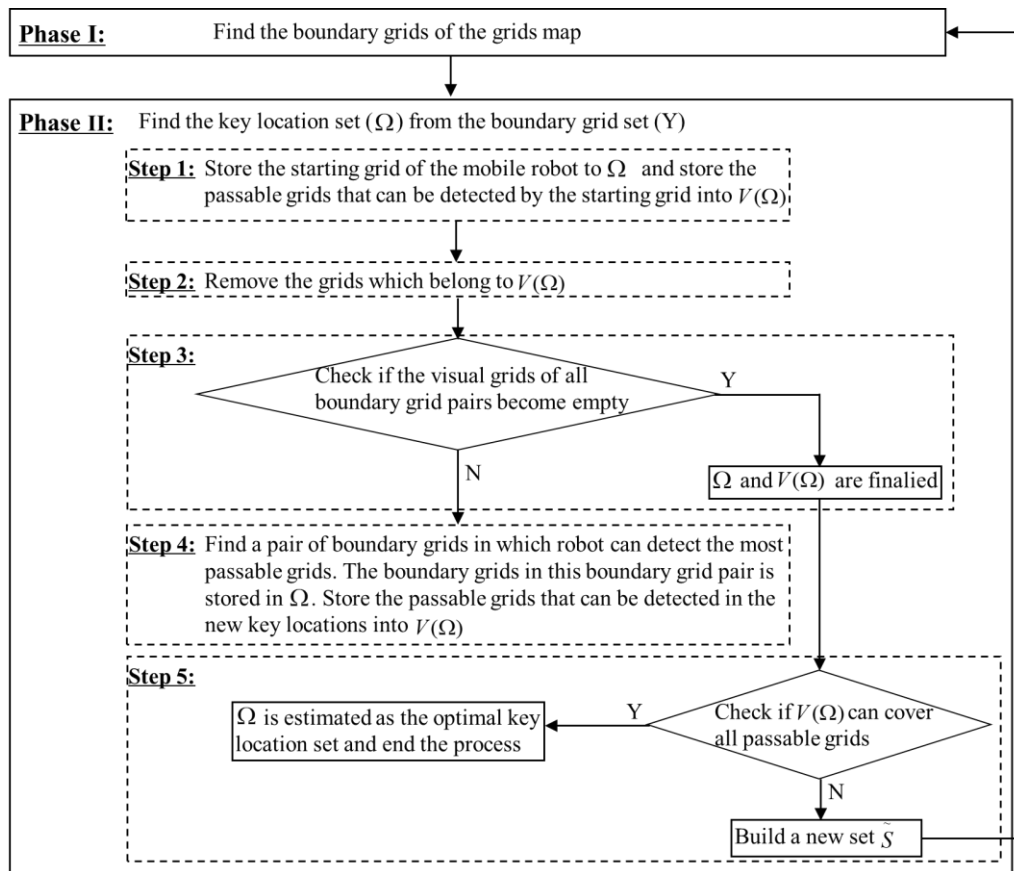


Figure 4. The flow chart of the method to pick out key locations in the grid map.

Phase I: Apply the pixel-level corner detection method to find the boundary grid set (Y) of the grids map.

$$\begin{cases} i \in Y & E(i) > 0 \\ i \notin Y & E(i) = 0 \end{cases} \quad i \in A \quad (15)$$

where:

$$E(i) = e(x_i, y_i) \cdot \left[\sum_u \sum_v (e(x_i + u, y_i + v) - e(x_i, y_i))^2 \right] \quad i \in S, u \in \{-1, 0, 1\}, v \in \{-1, 0, 1\} \quad (16)$$

$$e(x_i, y_i) = \begin{cases} 0 & i \in O \\ 1 & i \in S \end{cases} \quad (17)$$

As presented in Equation (15), if the boundary judgment parameter for grid i , i.e., $E(i)$ calculated by Equation (16), is greater than 0, the grid i is the boundary grid, i.e., $i \in Y$. A is the set of all identical

grids of the grid map. In Equation (16), x_i and y_i are the coordinates of grid i ; and $e(x_i, y_i)$ is the attribute value of grid i . The attribute value of obstacle grid (set O) is 0, and the attribute value of passable grid (set S) is 1, as formulated in Equation (17). $e(x_i + u, y_i + v)$ is the attribute value of surrounding grid of the grid i . u and v are the offsets of the abscissa and ordinate respectively, which belong to $\{-1, 0, 1\}$. Figure 5 presents a simple example.

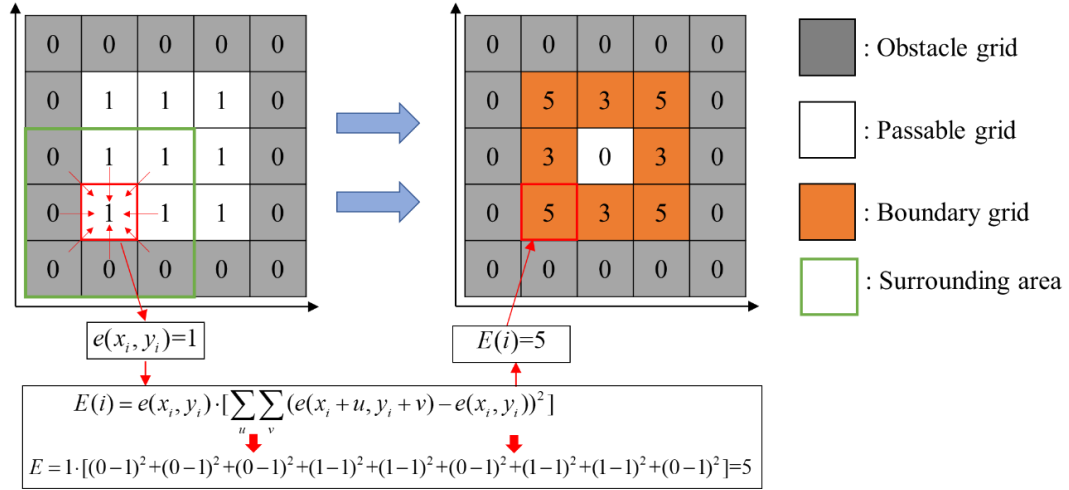


Figure 5. The calculation process example of the boundary judgment parameter for each grid.

Phase II: Find the key location set (Ω) from the boundary grid set (Y). As there are multiple boundary grids, the computational complexity of searching the key locations from all boundary grids is still high. The basic idea is to first form boundary grids into boundary pairs ($\{i, j\}, i, j \in Y$), and passable grids which can be detected by the boundary grids i and j are defined as the visible grids of boundary grid pair $v_{\{i,j\}}$, where $i, j \in Y$. Then, the pairs of boundary grids in which the robot can detect the most passable grids are stored into the key location set Ω . The detailed procedure is described in the following:

- Step 1: store the starting grid of the mobile robot to the key location set Ω ; and store the passable grids that can be detected by the starting grid into the visible grid set of the key locations, i.e., $V(\Omega)$.
- Step 2: remove the grids which belong to $V(\Omega)$ in the visual grids of all boundary grid pairs.
- Step 3: check if the visual grids of all boundary grid pairs become empty. If so, the key location set (Ω) and visible grid set $V(\Omega)$ are finalized, and go to Step 5. Otherwise, go to Step 4.
- Step 4: find a pair of boundary grids in which robot can detect the most passable grids. If there are multiple boundary grid pairs in which robot can detect the most passable grids, we select the boundary grid pair which is closest to the starting grid. The boundary grids in this boundary grid pair is stored in the key location set (Ω). Store the passable grids that can be detected in the new key locations into the visible grid set of key locations ($V(\Omega)$). Then, return to Step 2.
- Step 5: check if the visible grid set of key locations ($V(\Omega)$) can cover all passable grids. If so, the key location set (Ω) is estimated as the optimal key location set and end the process. If $V(\Omega)$ cannot cover all passable grids, we need to build a new set \tilde{S} . The new set \tilde{S} only retains these passable grids which cannot be detected in the original set S . Replace set S with set \tilde{S} and add these passable grids in the original set S that already be detected to set O . Then, repeat the Phase I.

An example of key location determination algorithm is detailed in Figure 6. Figure 6a presents a simple test map, where obstructed grids are in black. We first use the pixel-level corner detection method described in Phase I to find the boundary grids, and set the starting grid 7. According to the sign of the robot, the passable grids that can be detected by the starting grid are $\{7, 8, 12, 13\}$. Then, boundary grids are formed into boundary pairs ($\{i, j\}, i, j \in Y$), and passable grids which can be detected

by the boundary grids i and j are defined as the visible grid pair sets $v_{\{i,j\}}$, where $i, j \in Y$ (see Figure 6b). The procedure to find the key location set (Ω) from the boundary grid set is described in Figure 6c. The detailed procedure is described in the following steps:

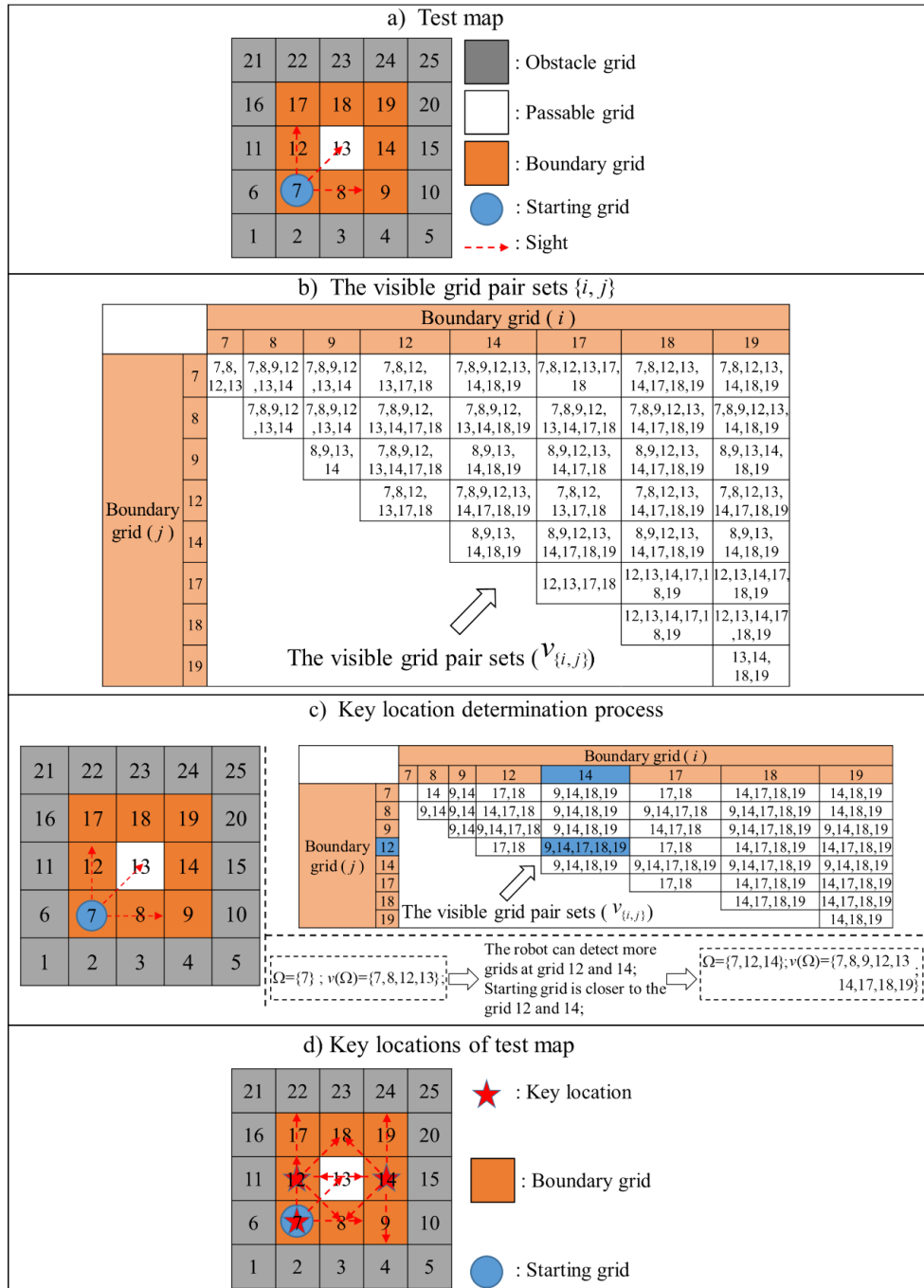


Figure 6. The repetitive process of key location determination.

- (1) Set the starting grid 7 to the key location set Ω ; and store the passable grids that can be detected ($\{7,8,12,13\}$) into the visible grid set of the key locations $V(\Omega)$.
- (2) Remove the grids belonging to $V(\Omega)$ in the visual grids of boundary grid pairs $v_{\{i,j\}}$.
- (3) If the visual grids of all boundary grid pairs are not empty, then go to next step.

- (4) Find some pair of boundary grids in which the robot can detect the most passable grids (i.e., {12,14}, {14,17}, {8,18}, {9,18}, {14,18}). Since {12,14} is the closest one, grids 12 and 14 are stored into the key location set (Ω). Then the new key locations set is ({9,14,17,18,19}).

Check if the visible grid set of key locations ($V(\Omega)$) can cover all passable grids. If cation set (Ω) is estimated as the optimal key location, set (i.e., {7,12,14}), as shown in Figure 6d.

4.2. Stage II: Ant Colony Optimization Algorithm (ACO)

Section 4.1 essentially provides the key locations set of the grid map Ω . Given the key locations and the distance between each pair of key locations, we then need to solve the shortest path for the robot to visit each key location from the starting location.

To solve this problem, we first use the A-star algorithm to determine the distance between two key locations in the grid map [15]. More importantly, to improve efficiency, we propose to apply the ACO to find a near-optimal solution. ACO has been widely applied in many fields (e.g., [16–18]).

The following explains the detailed calculation steps for ACO.

- Step 1: Initialization. Initialize the number of ants M , pheromone importance factor κ , pheromone concentrations τ_0 , importance factor of heuristic function λ , volatile degree of pheromone ρ , the total amount of pheromone released Q , the maximum number of iterations t_{\max} , and the initial number of iterations $t = 1$.
- Step 2: Construct the solution space. The ants are randomly placed at different starting locations. Each ant ($\theta(\theta = 1, \dots, M)$) determines the next location $\in \Omega$ to visit based on the probability $\chi_{i,j}^{\theta}(t)$ calculated by Equations (18) and (19) until all the ants have reached all locations $\in \Omega$.

$$\chi_{i,j}^{\theta}(t) = \begin{cases} \frac{[\tau_{i,j}(t)]^{\kappa} \cdot [\eta_{i,j}(t)]^{\lambda}}{\sum_{\varepsilon \in allow_{\theta}} [\tau_{i,\varepsilon}(t)]^{\kappa} \cdot [\eta_{i,\varepsilon}(t)]^{\lambda}}, & \varepsilon \in allow_{\theta} \\ 0, & \varepsilon \notin allow_{\theta} \end{cases} \quad (18)$$

$$\eta_{i,j}(t) = \frac{1}{d(i,j) \cdot f_j} \quad (19)$$

where:

$\chi_{i,j}^{\theta}(t)$ is the probability of ant θ transferring from location i to j at iteration t .

$\tau_{i,j}(t)$ is the pheromone concentration on the connection path between i and j at t .

$d(i,j)$ illustrates the distance of those key locations i and j .

f_i is the frequency of the robot which is unable to search for the target at the location j .

$allow_{\theta}$ is the set of locations to be visited for ant θ .

κ is the pheromone importance factor. The larger κ , the more significant the role of pheromone concentration in the path selection.

λ is the critical factor of heuristic function. The larger λ , the more significant the role of the heuristic function in the path selection. That is, ants will move to key locations with short distances with a high probability.

- Step 3: Determine the optimal solution. The path length of each ant $L_{\theta}(\theta = 1, \dots, M)$ will be calculated according to the distance between each key position, and all paths are recorded in the record table. Meanwhile, we record the optimal solution in the current number of iterations.
- Step 4: Update the pheromone. The pheromone concentration on the connection path between key locations is updated according to Equations (20)–(22).

$$\tau_{i,j}(t+1) = (1-\rho) \cdot \tau_{i,j}(t) + \Delta\tau_{i,j} \quad 0 < \rho < 1 \quad (20)$$

$$\Delta\tau_{i,j} = \sum_{\theta=1}^M \Delta\tau_{i,j}^{\theta} \quad (21)$$

$$\Delta\tau_{i,j}^{\theta} = \begin{cases} \frac{Q}{L_{\theta}} & \text{Ant } \theta \text{ transfers from } i \text{ to } j \\ 0 & \text{Otherwise} \end{cases} \quad (22)$$

where: Q represents the total amount of pheromone released, and it is constant. L_{θ} is the distance of the ant θ traverse path.

- Step 5: Judge the termination conditions. If the maximum iteration is met, stop calculation and output the optimal result. Otherwise, iteration increases by one, the records of the ant paths are cleared and return to Step 2.

5. Evaluation

5.1. Case Scenarios

We evaluated our algorithm on maps with grid sizes ranging from 10×10 , 20×20 , 30×30 cells (i.e., the environment is known), as shown in Figure 7. We used three different maps (cases) to verify our method. We define a grid of grid maps with a side length of 1 m and an area of 1 square meter. Therefore, these three cases are 100 square meters, 400 square meters, and 900 square meters, respectively.

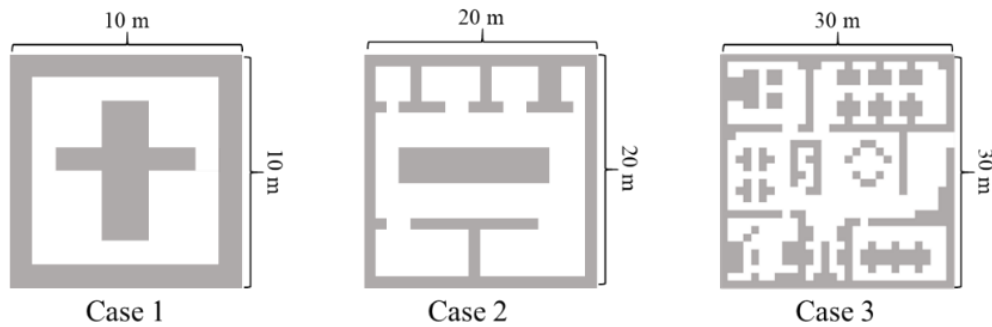


Figure 7. Case area.

The visual sensing parameter of the mobile robot (radius of circular measurement sectors) r is set as 15 m. The mobile robot will perform the target search process at a uniform speed (0.6 m/s) from the starting location of the map. The algorithm was run in an i7 processor @ 3.4 GHz, 32.00 GB RAM computer with a Windows 10 64bit operation system. For optimization, we used MATLAB R2015b.

Firstly, we assume no prior knowledge of target frequency, i.e., we do not have test data for searching the target in the working environment. In this simulation, a target is randomly placed in a passable grid of the grid map. Then we obtained the target search path through our optimization algorithm (OA), as shown in Figure 8. The total path length in case 1 brought by OA is 12 m. The time used in this target search in case 1 is 20 s. The total path length in case 2 brought by OA is 76 m. The time used in this target search in case 2 is 126.67 s. The total path length in case 3 brought by OA is 219 m. The time used in this target search in case 3 is 365 s.

5.2. Result Comparisons with Two Other Algorithms

We then compare our results with two other algorithms: Rule-based Algorithm (RBA) and Genetic Algorithm (GA) using the above three cases.

- (1) Rule-based Algorithm (RBA): Rule-based Algorithm is a heuristic algorithm, including the internal spiral coverage algorithm (ISC) for full coverage path planning of sweeping robot and the frontier-based algorithm for autonomous exploration of the robot, etc. The inner spiral coverage

algorithm was initially proposed by Butler et al. [19] based on the contact sensor algorithm. The basic idea of the internal spiral coverage algorithm is that the mobile robot advances clockwise or counterclockwise to traverse the entire work environment. The frontier-based algorithm for autonomous exploration of the robot was proposed by Brian et al. [20]. Frontiers are regions on the boundary between unexplored and explored space. To gain the newest and useful information, the robot must move to the borders and explore again.

- (2) Genetic Algorithm (GA): GA is an intelligent algorithm that was proposed by Holland in 1975 [21]. It is noted that GA has a strong global search ability and is suitable for discrete variable optimization problems. It imitates biological inheritance and evolution and introduces the concept of survival of the fittest in evolution into the algorithm.

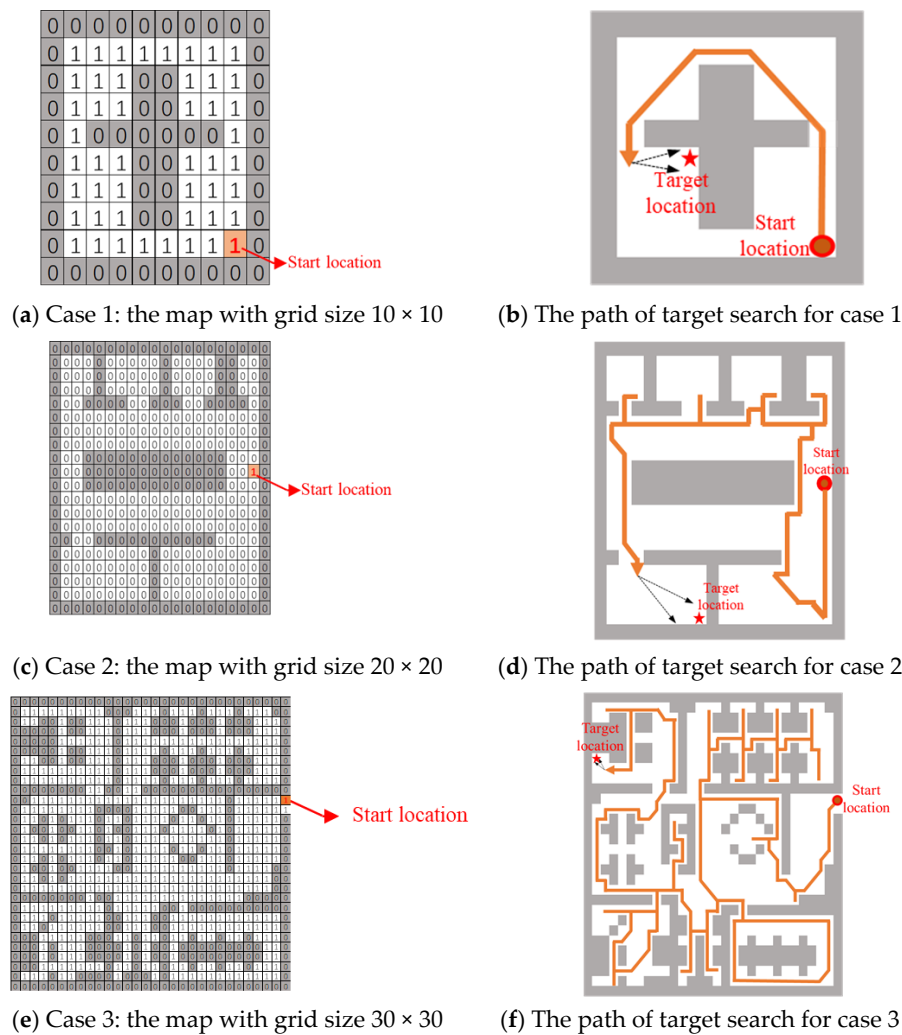


Figure 8. Generated path of target search for 3 cases.

5.2.1. Average Target Search Time

We set up 100 tests for each case. In each test, a target is randomly placed in a passable grid of the grid map. The mobile robot searches for the target at the same speed (0.6 m/s) according to the planned path obtained by RBA, GA, and OA. Then, we present the average time used in the process of target search in a total of 300 tests of 3 cases obtained by RBA, GA, and OA, as shown in Figure 9.

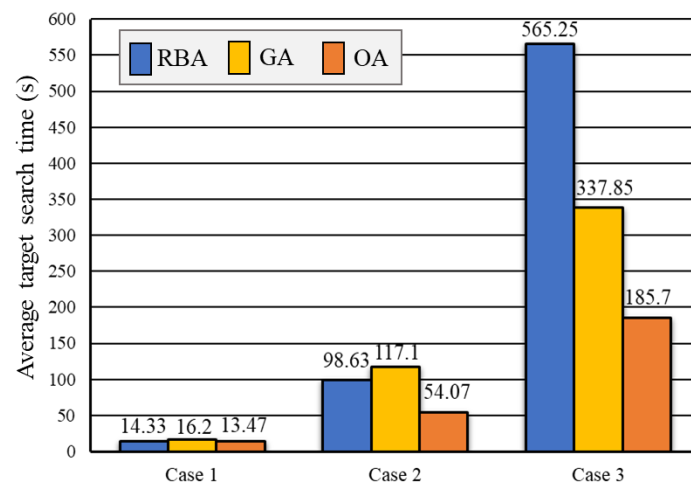


Figure 9. Average target search time for 3 cases obtained by Rule-based Algorithm (RBA), Genetic Algorithm (GA), and optimization algorithm (OA).

As shown in Figure 9, in Case 1, the average time obtained by RBA is 14.33 s, GA is 16.2 s, and our method OA is 13.47 s. In Case 2, the average time obtained by RBA is 98.63 s, GA is 117.1 s, and our method OA is 54.07 s. In Case 3, the average time obtained by RBA is 565.25 s, GA is 337.85 s, and OA is 185.7 s. The results show that the average time of target searching is significantly reduced with the help of our optimization algorithm. Note, the larger and more complex the environment map, the more pronounced the decline of the average time of target searching. The reduction rates of the average time of for three cases by three methods are also presented in Figure 10.

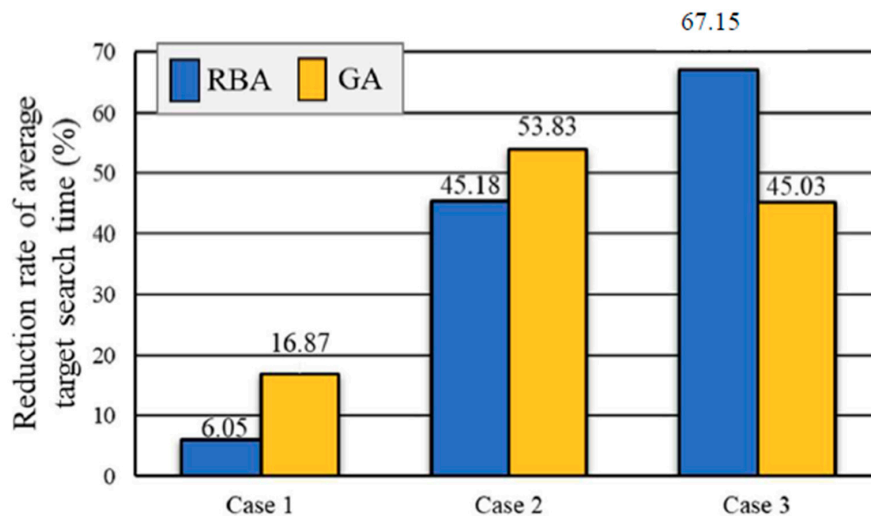


Figure 10. The reduction rate of average target search time by OA, RBA, and GA.

5.2.2. Path Length of Vision Complete Coverage of Passable Area

We present the total path length of 3 cases obtained by RBA, GA, and OA, as shown in Figure 11. The total path length in Case 1 brought by RBA is 21 m, by GA is 26 m, and by OA is 15 m. The total path length in Case 2 obtained by RBA is 100 m, by GA is 133 m, and by OA is 69 m. The total path length in the Case 3 obtained by RBA is 488 m, by GA is 376 m, and by OA is 231 m. The path distance reduction rates are shown in Figure 12. The comparisons show that the proposed OA algorithm can effectively reduce the length of full searching path.

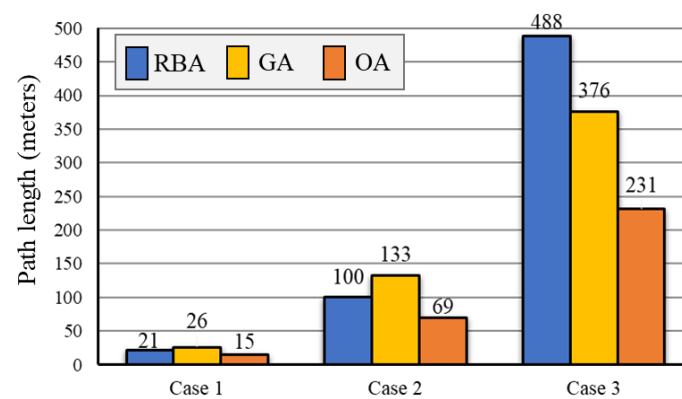


Figure 11. Path length of 3 cases obtained by RBA, GA, and OA.

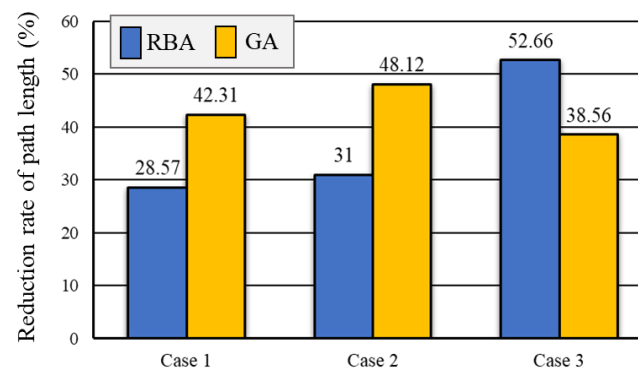


Figure 12. The distance reduction rate of path obtained by OA relative to RBA and GA.

5.3. Results with the Prior Knowledge of Target Frequency

Assuming the known prior knowledge of target frequency, we use the characteristics of the Poisson distribution to simulate the frequency of the target appearing. Poisson distribution is suitable for describing the number of random events occurring per unit time [22]. We set the probability of target occurrence in the 4 sub-regions based on the Poisson distribution (p -value is 0.226 obtained Kolmogorov–Smirnov (K-S) test). A p -value (calculated by SPSS) of 0.05 or less is commonly interpreted by statistician as justification for rejecting the null hypothesis that the data are Poisson distributed. As shown in Figure 13(a1–c1), we divide the passable area of the map into four different areas according to the frequency of target appearance.

The frequency of target non-occurrence is equal to 1 minus the probability of target occurrence. The optimization algorithm for path planning of target searching considering the target frequency is defined as the improved optimization algorithm (IOA). The original OA does not take into account the frequency of the target.

As shown in Figure 13, the path that can achieve vision complete coverage of passable areas obtained by OA is shorter than the path obtained by IOA.

We exhibit the path lengths, path length increase rate of IOA to OA, average time of target searching, time reduce rate of IOA to OA of 3 cases, as shown in Table 2.

Table 2. Path length (m), increase rate of path length (%), average time (s), and reduced rate of average time (%).

	Path Length (m)		Increase Rate of Path Length (%)	Average Time (s)		Reduce Rate of Average Time (%)
	OA	IOA		OA	IOA	
Case 1	15	18	20	10.71	9.7	9.43
Case 2	69	84	21.74	49.31	46.67	5.35
Case 3	231	234	1.3	177.49	143.24	19.3

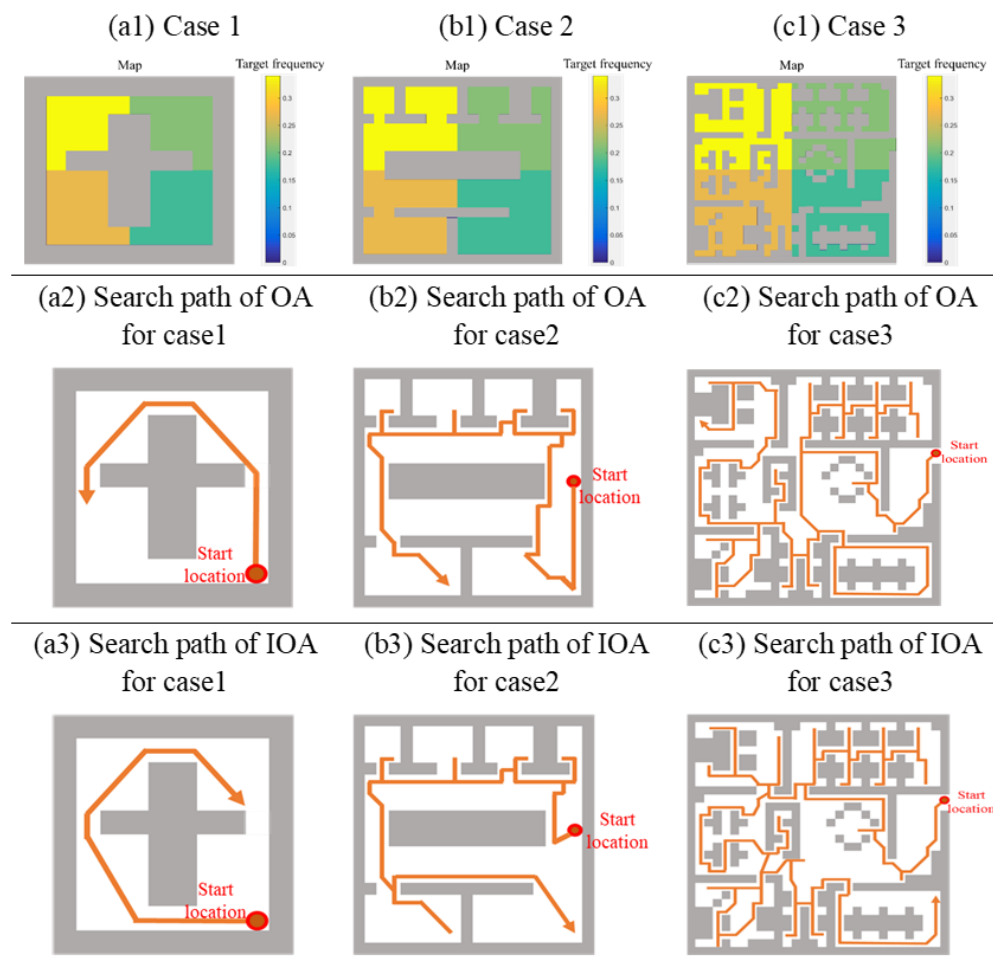


Figure 13. Planned paths based on an algorithm considering target frequency and not considering target frequency.

The results show that IOA can effectively reduce the average time of target searching (see Figure 14). The average time of target searching in case 1 for OA is 10.71 s, and 9.7 s for IOA, which is 9.43% shorter. The average time of target searching in case 2 for OA is 49.31 s, and 46.67 s for IOA, which is 5.35% shorter. The average time of target searching in case 3 for OA is 177.49 s, and 143.24 s for IOA, which is 19.3% shorter.

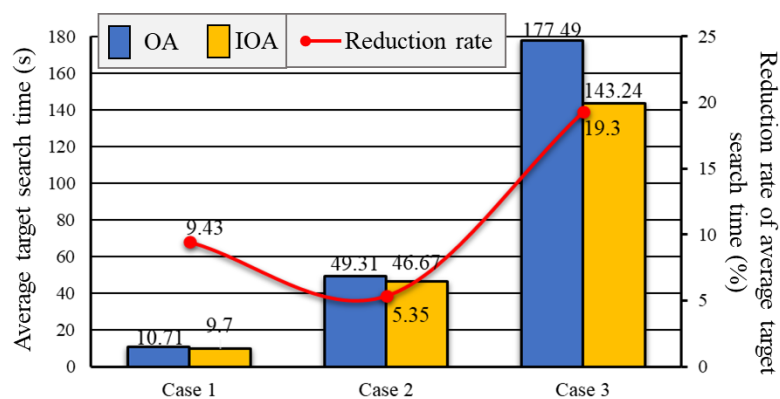


Figure 14. The distance reduction rate of path obtained by OA relative to RBA and GA.

5.4. Results of the Test in a Real Environment with a Real Robot

We further conduct a test in a real environment with a real robot (see Figure 15). The test environment is a corridor with many offices and some obstacles. A target exists in one place in the corridor. The test environment is about 576 square meters. The robot set off at one end of the corridor.

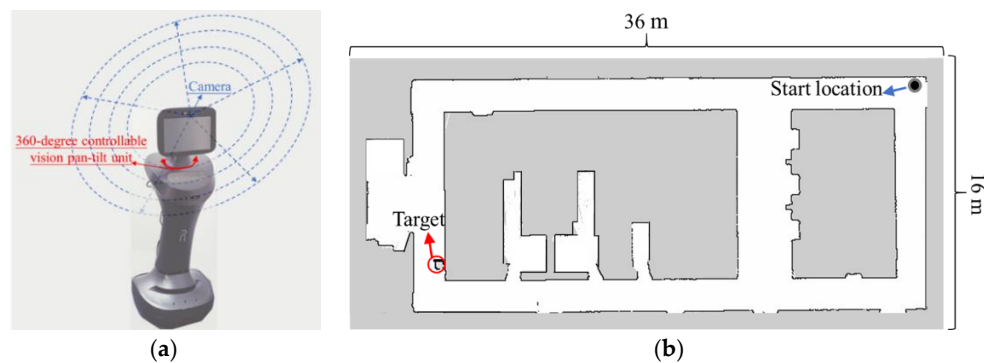


Figure 15. (a) A mobile robot for target searching; (b) a real test environment.

The target search path is obtained through our optimization algorithm in the real test environment, as shown in Figure 16. The target search path brought by OA is 57.7 m. The time used in this target search is 115 s. This test with a real environment using a real robot further verifies the feasibility of our algorithm.

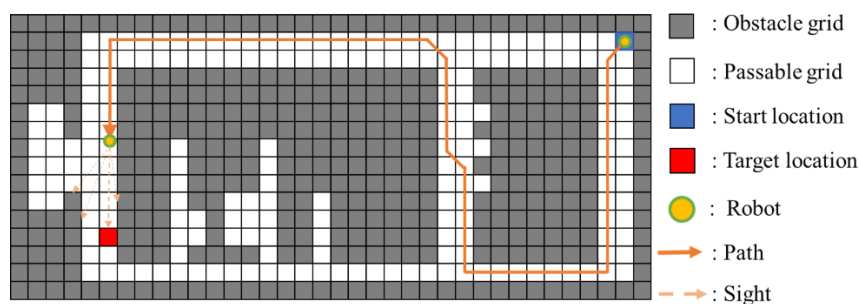


Figure 16. The distance reduction rate of path obtained by OA relative to RBA and GA.

6. Conclusions

This paper tackles the challenging path planning problem of target search for mobile robots. We propose a two-stage optimization model to efficiently obtain an approximate optimal solution. In this model, we first develop an efficient method to determine the key locations for visual complete coverage of a two-dimensional grid map, which is constructed by drawing lessons from the method of corner detection in the image processing. As the field of view of the robot camera can completely search the entire known area, the union of the area covered by the visual field of the robot camera at the key locations can achieve full coverage of the whole environment. Then, we design a planning problem for searching the shortest path that passes all key locations considering the frequency of target occurrence. Several tests are conducted in the paper to validate the proposed sufficient method to path planning problem of target search with a mobile robot. The results show that the proposed optimization algorithm (OA) can achieve the significantly shorter search path length and the shorter target search time than the current Rule-based Algorithm (RBA) and Genetic Algorithm (GA) in various simulation cases. Furthermore, the results show that the improved optimization algorithm (IOA) with the priori known frequency of occurrence of the target can effectively search the target with shorter time compared to OA. We also set up a test in a real environment to verify the feasibility of our algorithm.

The ultimate goal of our research was to improve the mobile robots' target searching in a complex environment with uncertainties from several elements and the dynamic obstacles. To achieve this

goal, first, we needed to plan the “global” optimal path for the static environment; then, the robot decides some “local” optimal path when detecting dynamic obstacles along the global path in real time. This paper only focuses on the first part of this problem since even for this problem there are many challenging technical problems which need to be addressed. After the robot detects the dynamic obstacles in real time, the robot will plan a local path to “pass” the dynamic obstacles and then return to the global path. This part is also a challenging research topic. In the future, we will introduce a more realistic and complicated situation which considers complex dynamic environment. In addition, due to lack of multiple test data for searching the target in the working environment (no prior knowledge of target frequency), we only conducted a simple and rough treatment for the frequency of target occurrence in our testing. Verifications using more actual test data in the future could further improve our work.

Author Contributions: Conceptualization, T.S. and X.W.; methodology, T.S. and X.H.; software, T.S.; validation, T.S., and X.H.; formal analysis, T.S. and X.W.; investigation, T.S., and X.H.; resources, T.S., and X.H.; data curation, T.S., X.H. and X.W.; writing—original draft preparation, T.S. and X.W.; writing—review and editing, T.S., X.H. and X.W.; visualization, T.S. and X.W.; supervision, T.S. and X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lakshmanan, A.K.; Mohan, R.E.; Ramalingam, B.; Le, A.V.; Ilyas, M. Complete coverage path planning using reinforcement learning for tetromino based cleaning and maintenance robot. *Autom. Constr.* **2020**, *112*, 103078. [\[CrossRef\]](#)
2. Yakoubi, A.M.; Laskri, M.T. The path planning of cleaner robot for coverage region using Genetic Algorithms. *J. Innov. Digit. Ecosyst.* **2016**, *3*, 37–43. [\[CrossRef\]](#)
3. Wang, Z.; Li, H.; Zhang, X. Construction waste recycling robot for nails and screws: Computer vision technology and neural network approach. *Autom. Constr.* **2019**, *97*, 220–228. [\[CrossRef\]](#)
4. Kim, J. Underwater surface scan utilizing an unmanned underwater vehicle with sampled range information. *Ocean Eng.* **2020**, *207*, 107345. [\[CrossRef\]](#)
5. Sandamurthy, K.; Ramanujam, K. A hybrid weed optimized coverage path planning technique for autonomous harvesting in cashew orchards. *Inf. Process. Agric.* **2019**, *7*, 152–164. [\[CrossRef\]](#)
6. Ellefsen, K.O.; Lepikson, H.A.; Albiez, J.C. Multiobjective coverage path planning: Enabling automated inspection of complex, real-world structures. *Appl. Soft Comput.* **2019**, *61*, 264–282. [\[CrossRef\]](#)
7. Glorieux, E.; Franciosa, P.; Ceglarek, D. Coverage path planning with targetted viewpoint sampling for robotic free-form surface inspection. *Robot. Comput. Integr. Manuf.* **2020**, *61*, 101843.1–101843.11. [\[CrossRef\]](#)
8. Galceran, E.; Carreras, M. Planning coverage paths on bathymetric maps for in-detail inspection of the ocean floor. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013.
9. Mansouri, S.S.; Kanellakis, C.; Georgoulas, G.; Kominiak, D.; Gustafsson, T.; Nikolakopoulos, G. 2D Visual Area Coverage and Path Planning Coupled with Camera Footprints. *Control Eng. Pract.* **2018**, *75*, 1–16. [\[CrossRef\]](#)
10. Wang, P.; Krishnamurti, R.; Gupta, K. View Planning Problem with Combined View and Traveling Cost. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2017.
11. An, W.; Shao, F.M.; Meng, H. The coverage-control optimization in sensor network subject to sensing area. *Comput. Math. Appl.* **2009**, *57*, 529–539. [\[CrossRef\]](#)
12. Angella, F.; Reithler, L.; Gallesio, F. Optimal Development of Cameras for Video Surveillance Systems. In Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), London, UK, 5–7 September 2007; pp. 388–392.

13. Arain, M.A.; Cirillo, M.; Bennetts, V.H.; Schaffernicht, E.; Lilienthal, A.J. Efficient Measurement Planning for Remote Gas Sensing with Mobile Robots. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 3428–3434.
14. Applegate, D.L. *The Traveling Salesman Problem: A Computational Study*; Princeton University Press: Princeton, NJ, USA, 2006.
15. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1986**, *4*, 100–107. [[CrossRef](#)]
16. Majeed, A.; Lee, S. A New Coverage Flight Path Planning Algorithm Based on Footprint Sweep Fitting for Unmanned Aerial Vehicle Navigation in Urban Environments. *Appl. Sci.* **2019**, *9*, 1470. [[CrossRef](#)]
17. Rueda, R.; Ruiz, L.G.B.; Cuellar, M.P.; Pegalajar, M.C. An Ant Colony Optimization approach for symbolic regression using Straight Line Programs. Application to energy consumption modelling. *Int. J. Approx. Reason.* **2020**, *121*, 23–38. [[CrossRef](#)]
18. Kumar, R.; Singh, R.; Ashfaq, H. Stability enhancement of multi-machine power systems using Ant colony optimization-based static Synchronous Compensator. *Comput. Electr. Eng.* **2020**, *83*, UNSP 106589. [[CrossRef](#)]
19. Butler, Z.; Rizzi, A.; Hollis, R. Contact sensor-based coverage of coverage of rectilinear environments. In Proceedings of the IEEE International Symposium on Intelligent Control Intelligent Systems and Semiotics, Cambridge, MA, USA, 15–17 September 1999; pp. 266–271.
20. Yamauchi, B. A frontier-based approach for autonomous exploration. In Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation, Monterey, CA, USA, 10–11 July 1997.
21. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; MIT Press: Cambridge, MA, USA, 1992.
22. Tomioka, Y.; Takara, A.; Kitazawa, H. Generation of an Optimum Patrol Course for Mobile Surveillance Camera. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 216–224. [[CrossRef](#)]

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).