Autonomous Robotic Exploration Based on Frontier Point Optimization and Multistep Path Planning

Article *in* IEEE Access · April 2019

DOI: 10.1109/ACCESS.2019.2909307

3 authors, including:

Baofu Fang
Hefei University of Technology
55 PUBLICATIONS **290** CITATIONS

SEE PROFILE

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 8, 2019, accepted March 31, 2019. Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2019.2909307

Autonomous Robotic Exploration Based on Frontier Point Optimization and Multistep Path Planning

BAOFU FANG[1], JIANFENG DING[1], AND ZAIJUN WANG[2]
[1]School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China
[2]CAAC Academy of Flight Technology and Safety, Civil Aviation Flight University of China, Guanghan 618307, China

Corresponding author: Baofu Fang (fangbf@hfut.edu.cn)

This work was supported in part by the Natural Science Foundation of Anhui Province under Grant 1708085MF146, in part by the Project of Innovation Team of Ministry of Education of China under Grant IRT17R32, and in part by the Open Foundation from the CAAC Academy of Flight Technology and Safety under Grant F2018KF06.

ABSTRACT Autonomous robotic exploration of an unknown environment is a key technology for robot intelligence. In order to improve the efficiency of it, we propose a strategy based on frontier point optimization and multistep path planning in this paper. In the frontier points' optimization section, we present a random frontier points' optimization (RFPO) algorithm to select the frontier point with the highest evaluation value as the target frontier point. The evaluation function of frontier points is defined by considering information gain, navigation cost, and the precision of the localization of the robots. In the path planning section, we propose a multistep exploration strategy. Instead of planning the global path from the current position of the robot to the target frontier point directly, we set a local exploration path step size. When the robot's movement distance reaches the local exploration path step size, we reselect the current optimal frontier point for path planning to reduce the possibility that the robot may take some repetitive paths. Finally, the relevant experiments are carried out to verify the effectiveness of this strategy.

INDEX TERMS Autonomous exploration, random frontier points optimization algorithm, frontier point evaluation function, multistep path planning.

I. INTRODUCTION

Autonomous robotic exploration [1], [2] is a major research issue in robotics. The primary goal is to make robots to acquire the most complete and accurate map of an environment in a finite time without human intervention. Many existing map exploration strategies are based on frontier [3]–[7], [9], which is the boundary between unexplored space and known space. The idea of frontier-based exploration strategy is to direct robots to unknown regions to complete exploration missions, thus the autonomous exploration task can be divided into three general steps: generation of frontier points, selection of the frontier point with the highest evaluation value, path planning to the selected frontier point.

Generation of frontier points is the premise of frontier-based exploration strategy. In existing research, some frontier points generation algorithms are based on the edge detection and region extraction of digital image processing technology [3], [4]. In order to extract frontier edges, the entire map has to be processed, and as the map expands, processing it will consume more and more computational resources. This has led to research on efficient detection of frontier edges. Keidar and Kaminka [5] proposes an algorithm for frontier detection based on processing only the new laser reading data. In the study by Senarathne *et al.* [6], he presents an approach to generate frontier points by tracking intermediate changes to grid cells and considering only the updated grid cells for the final frontier generation operation. Umari and Mukhopadhyay [8] apply the Rapidly-exploring Random Tree (RRT) algorithm to frontier points generation. Due to the randomness of the RRT algorithm, the generated frontier points are unevenly distributed.

The selection of target frontier points is the key to efficient exploration. Frontier-based strategy is first introduced by Yamauchi [9]. The exploration strategy used is to identify all the frontier regions in the current map and then

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

drive robots to the nearest frontier point. This method has two shortcomings for the exploration task. First, it treats all frontiers equally. Secondly, it is limited to one source of information: finding new terrain. So many different frontier points selection algorithms are proposed. Simmons *et al.* [10] and Moorehead *et al.* [11] present the frontier point selection function by combining information gain and exploration cost to select the target frontier point. Carlone and Lyons [12] uses the mixed-integer linear programming (MILP) model to obtain the optimal frontier point for autonomous exploration. The work by Mei *et al.* [13] proposes an algorithm to choose the next target frontier point for the robot to explore based on orientation information. The team of Laguna university and Bonn university [14] propose a novel exploration strategy that exploits background knowledge by considering previously seen environments to make better exploration decisions. Gautam *et al.* [15] uses K-means algorithm to cluster frontier points and assigns these frontier points to the robots using a Hungarian method.

In the path planning section, Bircher *et al.* [16] and Ellips and Hossein [17] present an algorithm which generate a search path for robots based on the RRT algorithm. Lauri and Ritala [18] formulate the problem as a partially observable Markov decision process (POMDP) with an information-theoretic objective function, and solve it applying forward simulation algorithms with an open-loop approximation. By sampling the local environmental information, the forward simulation is performed to calculate different information gains of each path to determine the choice of the exploration path. Stachniss *et al.* [19] proposes an algorithm which uses a highly efficient Rao-Blackwellized particle filter to represent the posterior about maps and poses. It trades off the cost of executing an action with the expected information gain and takes into account possible sensor measurements gathered along the path taken by the robot. Sometimes, robots will go to the place where they have gone before to reduce the uncertainty. In the study by Elhoseny *et al.* [20], he proposes a Genetic Algorithm (GA) based path planning method in order to work in dynamic and complex environments with obstacles. In the study by Senarathne *et al.* [21], he augments the traditional frontier-based exploration strategy to include a probabilistic decision step that decides whether further motion on the planned path to the next sensing location is desirable or not. If the motion is not desirable, it is cancelled and a new sensing location is selected as the next sensing task.

In this paper, we present a strategy based on frontier points optimization and multistep path planning. In order to get the best frontier point, we propose a Random Frontier Points Optimization (RFPO) algorithm. This algorithm optimizes the random frontier points generated by the RRT algorithm. Combining this algorithm with the frontier points evaluation function, we can obtain the current optimal frontier point. And one primary problem is how to evaluate the frontier points. In this paper, we consider three factors: information gain at the frontier point, navigation cost and the accuracy of robot positioning. Taking these factors into account, we define a frontier points evaluation function to evaluate all frontier points. In the process of robot's path planning to the optimal frontier point, we propose the strategy of multistep exploration. According to the map size, we define a local exploration path step size. When the robot's movement distance reaches the local exploration path step size, the current optimal frontier point is recalculated and reselected for exploration. This may prevent the robot from taking some repetitive paths.

The organization of this paper is as follows. Section II introduces the optimal frontier point extraction, including three parts: generation of frontier points, frontier points evaluation function, RFPO algorithm. Section III describes the strategy of multistep exploration. In section IV, we carry out relevant experiments to verify the effectiveness of our strategy. Finally, the paper is concluded in Section V.

II. OPTIMAL FRONTIER POINT EXTRACTION

A. GENERATION OF FRONTIER POINTS

In this paper, we use the SLAM algorithm GMapping [25], [26] to build a 2D occupancy grid map. In the occupancy grid map, grid cell has three states: free, occupied and unknown. Frontier points in an occupancy grid map are defined as the boundaries between grids categorized as free and unknown. We generate frontier points in the map by using the RRT [27] algorithm. The advantage of this algorithm is that the spanning tree is simple to build and it can quickly traverse unexplored areas in the space, which is especially suitable for systems that contain obstacles. In addition, for a closed environment, the RRT algorithm provides completeness, which can guarantee that the robot will explore all regions and build a complete map in the process of autonomous exploration.

At first, initializing the rapidly exploring random tree. Once the tree is initialized, we insert the current position of the robot $p_{current}$ as the root node of the tree. We denote the frontier points as p_i, where i is the subscript to distinguish different frontier points. The new point p_{rand} is generated randomly on the map. Find out the closest node $p_{nearest}$ that already exists in the current tree to the newly generated point. Connect these two points to get a straight line, move a length of η from point $p_{nearest}$ to point p_{rand} alone this line to get a new point p_{new}. η represents the growth rate of the tree. The tree will grow quickly when η is set large, but it will not reach some small corners. On the contrary, when η is set relatively small, it can reach the corner but the speed of generating frontier points will decrease. In order to maintain the balance between the two factors, we set it according to the map size. If there are no obstacles on the lines between $p_{nearest}$ and p_{new}, the corresponding points and edges are added to the tree. Otherwise, the generated point is discarded. At the same time, if the newly generated point lies in the unknown region, the point is considered as a frontier point. It will be marked in the map, and then we stop the growth of this tree. Using

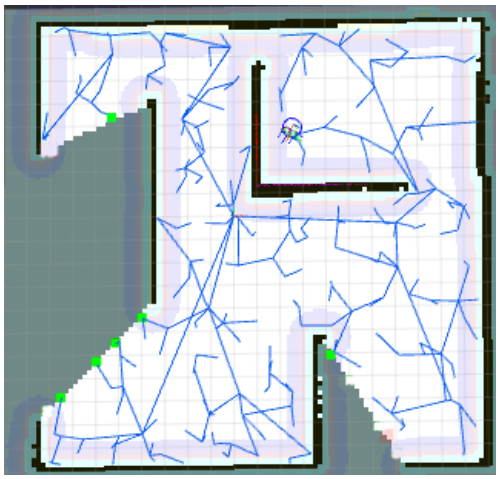


FIGURE 1. Generation of frontier points. White areas represent free space, gray areas represent unknown region, black areas represent obstacle, blue lines represent spanning tree, green points represent frontier points.

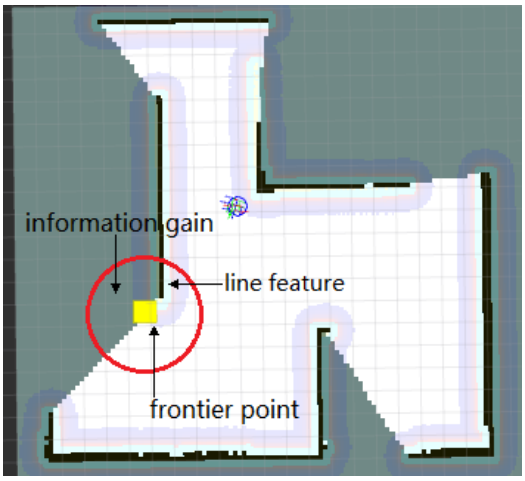


FIGURE 2. Frontier point detection circle. The yellow point represents frontier point, the gray area in the circle represents the information gain.

the current robot's position as the new root node, we build a new rapidly exploring random tree to generate frontier points. One example of frontier points generation is shown in Fig.1.

B. FRONTIER POINTS EVALUATION FUNCTION

Frontier points evaluation function is the basis for the selection of frontier points. We evaluate the frontier points from the following three factors: information gain at the frontier point, navigation cost and the precision of the localization of the robots. Among these factors, information gain is defined as the area of unknown region expected to be explored for a given frontier point. At present, the calculation method of information gain can be divided into two categories: one is to measure the undetected space size in the visible region of the target frontier point directly [22], and the other is based on the information entropy method, which was first proposed by Bourgault [23], [24]. In this paper, we calculate the information gain by the first method. Taking the frontier point as the circle center, we form a circle with the radius of the laser's detection distance. Frontier point detection circle is shown in Fig.2. The information gain is quantified by counting the number of unknown cells in this circle. Navigation cost is defined as the expected distance to be traveled by a robot to reach a frontier point. The Euclidean distance from the robot's current position to the target frontier point is calculated to represent it. In addition, the accurate map depends on the robot's accurate estimate of its own pose. And the robot can locate itself more accurately if there are more line features or other features (such as breakpoint, corner, polylines) can be detected within the detection range of the target frontier point. In order to be consistent with the calculation unit of information gain, we use the area of obstacles in the frontier point detection circle to represent it. It is quantified by counting the number of occupied cells in the circle. Based on the above factors, we assume that when the robot has reached the target frontier point p_i at time t, the information gain is I_t, the navigation cost is C_t, the area of obstacles is F_t, and the following frontier points evaluation function is defined:

$$E\left(p_i(t)\right) = \frac{\alpha\left(I_t - I_{t-1}\right) + \gamma\left(F_t - F_{t-1}\right)}{\beta\left(C_t - C_{t-1}\right)} \tag{1}$$

where α, β, γ are the weights of information gain, navigation cost and the area of obstacles. These weights are used to adjust the importance of different factors. Its value can be set according to different tasks and environments. If the exploration task requires the exploration to be completed as quickly as possible, increase the value of α and decrease the value of γ; if the exploration task focuses more on the accuracy of the map, decrease the value of α and increase the value of γ. β usually takes 1 to represent the gain value under the unit navigation cost. We use the frontier points evaluation function to evaluate all the frontier points. The point with the highest value is selected as the target frontier point.

C. RANDOM FRONTIER POINTS OPTIMIZATION ALGORITHM

Because the generation of frontier points part is always running throughout the exploration process, we will get many frontier points as the exploration task is executed. However, due to the randomness of the RRT algorithm, the distribution of these frontier points is uneven. Therefore, we need to optimize the generated frontier points. Borrowing the idea of GSO [29], [29] algorithm, we propose the RFPO algorithm. The GSO algorithm is a new type of bionic swarm intelligent optimization algorithm. It simulates the natural phenomenon that the fireflies with high luminance values will attract fireflies with low luminance values to move to it, making all fireflies concentrated in a better position so as to realize optimization of the problem.

In the RFPO algorithm, we consider each frontier point as a firefly and use the value of the frontier points evaluation function $E\left(p_i(t)\right)$ as its absolute brightness value $L_i(t)$:

$$L_i(t) = E(p_i(t)) \tag{2}$$

If the absolute brightness value of firefly i is greater than the absolute brightness value of firefly j, the firefly j will be attracted by firefly i and moves to it. The magnitude of this attraction is determined by the relative brightness value of firefly i to firefly j. The brightness intensity of firefly i at the location of firefly j is defined as the relative brightness of firefly i to firefly j. The greater the relative brightness value is, the greater the attraction is. Therefore, in order to model the attraction of frontier point i to frontier point j, we must first model the relative brightness of frontier point i to frontier point j. Considering that the relative brightness of firefly decreases with distance increasing, the relative brightness value of frontier point i to frontier point j is defined as:

$$L_{ij}(t) = L_i(t)e^{-r_{ij}^2} \tag{3}$$

where r_{ij} is the distance from frontier point i to frontier point j. Assuming that the attraction of firefly i to firefly j is proportional to the relative brightness of firefly i to firefly j, the attraction of frontier point i to frontier point j can be calculated by (4) below:

$$A_{ij}(t) = \rho L_{ij}(t) \tag{4}$$

where ρ is the coefficient of attraction and for all frontier points, the different value of ρ have the same effect, so we can take $\rho = 1$.

For each frontier point, there is a perception radius r. Its value should be set according to the perception sensor range. In this range, every frontier point will find the other frontier points whose absolute brightness value is larger than themselves to form their own neighborhood set $N_i(t)$:

$$N_i(t) = \{r_{ij} \le r; L_i(t) < L_j(t)\} \tag{5}$$

After determining the neighborhood set, the probability that the frontier point moves to other frontier points in its neighborhood set is calculated as follow:

$$P_{ij}(t) = \frac{A_{ij}(t)}{\sum_{k \in N_i(t)} A_{ik}(t)} \tag{6}$$

In order to select a target point to move to in the neighborhood set, we use the method of roulette. The roulette is made [30] based on the probability values calculated above. Once the target frontier point p_j has been determined, the movement of the frontier point p_i is calculated by the following (7):

$$p_i(t+1) = p_i(t) + s\left(\frac{p_j(t) - p_i(t)}{\|p_j(t) - p_i(t)\|}\right) \tag{7}$$

where $p_i(t), p_j(t)$ represent the current position of the frontier point, and $p_i(t+1)$ is the position of the frontier point i after the movement. s represents the movement step size, and its value can be determined according to the sensor measurement range. Setting the iteration variable m, we will get one or more local convergence points when the optimization iteration ends.

The generation of frontier points and the movement of the robot are performed simultaneously. When the robot's movement distance s_{move} is less than the local exploration path step size s_{fixed}, the RRT algorithm is working to generate frontier points. Once the robot's movement distance s_{move} reaches the local exploration path step size s_{fixed}, we remove the frontier points that are no longer at the frontier and the frontier points that robot cannot arrive. Then we optimize all the frontier points and select the frontier point with the highest evaluation value from the optimization result as the target frontier point. Fig.5 in section IV shows the effect of optimization. The entire procedure for extracting the optimal frontier point is described in Algorithm 1:

III. MULTISTEP EXPLORATION STRATEGY

Path planning is also crucial to the efficiency of autonomous exploration. In this paper we propose a multistep exploration strategy. After obtaining the current optimal target frontier point according to the above steps, we do not plan the global path from the current position of the robot to the optimal target frontier point directly. Instead, we define a local exploration path step size s_{fixed} for multistep path planning. The reason why we define s_{fixed} is that: in the process of robot's movement, some new frontier points will be generated, some old frontier points will become invalid, and the new generated frontier point may be superior to the current optimal target frontier point. This may cause the robot to take some repetitive paths. Therefore, we define a local exploration path step size. Each time when the movement distance of the robot reaches the step size, we clear the invalid points, and all the remaining frontier points are reoptimized and reselected so as to avoid this situation from happening. Within each local exploration path step size, we use the dynamic approach [32]. The dynamic window approach is used for robot local path planning to avoid obstacle.

In the dynamic window approach, it is necessary to know the motion model of the robot in order to simulate the corresponding motion trajectory of the robot according to the given velocity. In this paper, we use the differential drive robot which has linear velocity and angular velocity. Assuming that at time t the pose of the robot is $[x_t, y_t, \theta_t]^T$, the given velocity pair is $[v_t, w_t]^T$, and the window time is Δt. Then the calculation model is as follows:

$$x_{t+1} = x_t - \frac{v_t}{w_t}\sin\theta_t + \frac{v_t}{w_t}\sin(\theta_t + w_t\Delta t) \tag{8}$$

$$y_{t+1} = y_t - \frac{v_t}{w_t}\cos\theta_t - \frac{v_t}{w_t}\cos(\theta_t + w_t\Delta t) \tag{9}$$

$$\theta_{t+1} = \theta_t + w_t\Delta t \tag{10}$$

By the above robot motion model, we can calculate the corresponding motion trajectory according to different velocities. We choose multiple sets of different velocities within a limited range according to the robot's own velocity limit and environmental constraints, and simulate the robot's motion trajectory. Then we redefined the trajectory evaluation function to evaluate the multiple generated trajectories, and select

Algorithm 1 Optimal Frontier Points Extraction

Input: map
Output: goalpoint
tree ← InitialzeTree (η)
tree ← InsertNode ($p_{current}$, $\emptyset$)
InitializeGSO (ρ, r, m, s)
while True **do**
  while $s_{move} < s_{fixed}$
    p_{rand} ← Random
    $p_{nearest}$ ← Nearest($tree$, p_{rand})
    p_{new} ← Move($p_{nearest}$, p_{rand}, η)
    if ObstacleFree($p_{nearest}$, p_{new}) **then**
      $tree$ ← InsertNode(p_{new},($p_{nearest}$, p_{new}))
      if UnknowRegion(p_{new}) **then**
        PublishPoint(p_{new})
        $tree$ ← InsertNode($p_{current}$, $\emptyset$)
      end if
    end if
  end while
  Clear Valid Points
  if Number(p) = 0 **then**
    break
  end if
  set$t = 1$
  for $t \leq m$ **do**
    for $i = 1$ to N**do**
      $L_i(t) = E(p_i(t))$
    end for
    for $i = 1$ to N**do**
      $N_i(t) = \{r_{ij} \leq r; L_i(t) < L_j(t)$
      for each $j \epsilon N_i(t)$ **do**
        $L_{ij}(t) = L_i(t)\, e^{-r_{ij}^2}$
        $A_{ij}(t) = \rho L_{ij}(t)$
        $P_{ij}(t) = \frac{A_{ij}(t)}{\sum_{k \in N_i(t)} A_{ik}(t)}$
      end for
      j = Roulette($P_{ik}(t), k \in N_i(t)$)
      $p_i(t+1) = p_i(t) + s\left(\frac{p_j(t) - p_i(t)}{\|p_j(t) - p_i(t)\|}\right)$
    end for
    $t = t + 1$
  end for
  return goalpoint = Max($E(p)$)
end while

the trajectory with the highest score to be performed by the robot. Considering the following factors in the process of autonomous exploration and map building: new information gain ΔI, new observed area of obstacles ΔF, angular deviation $\Delta\theta$ and distance Δl between simulated pose and target pose, the nearest distance Δd between the simulated pose and the obstacle, we define the following evaluation function:

$$R(v, w) = \varepsilon\Delta I + \varphi\Delta F - \phi\Delta\theta - \mu\Delta l + \sigma\Delta d \quad (11)$$

where ε, φ, ϕ, μ, σ are the weights of the above factors. In order to eliminate the influence of different calculation units in (11), each part must be normalized first. For example, we normalize the nearest distance between the simulated pose and the obstacle as follow:

$$normal\_d(i) = \frac{\Delta d(i)}{\sum_{i=1}^{n} \Delta d(i)} \quad (12)$$

where i represents the trajectory to be evaluated, n is all the trajectories we generated by sampling.

The physical meaning of the evaluation function is: in the process of the robot's local exploration and navigation, the robot is required to avoid obstacles in real time and arrive the target frontier point as quickly as possible, and more unknown regions can be explored along the motion trajectories. For robots, the features such as lines, breakpoint, corner, polylines can help it to locate more accurately. Therefore, the more features that can be observed along the trajectories of the robot, the more accurately the robot will locate itself, and the map will be more accurate. As shown in the Fig.3 below, the yellow point is the target frontier point. We simulate a number of trajectories at different velocity sets (only some of the trajectories are shown in the Fig.3). According to our defined trajectory evaluation function, we select the trajectory with the highest score, it is marked in red in the following Fig.3. It can be seen that the robot performs the red trajectory can reach the target frontier point quickly. At the same time, the trajectory has a certain safety distance from the wall, and the wall border can help the robot to locate itself more accurately.

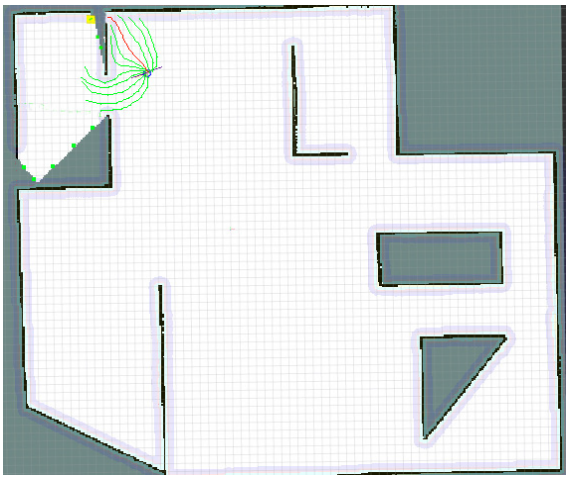


FIGURE 3. Selection of optimal exploration trajectory.

IV. EXPERIMENTS AND RESULTS

A. EXPERIMENTAL SETUP

Experiments are carried out to evaluate the performance of the strategy that we propose using the simulation map and the real map. The results are compared with other strategies. All strategies used for comparisons are developed as ROS components [33] in C++ using ROS libraries on a computer with Intel core i7 3.60GHz processor and 8GB of RAM running Ubuntu 14.04. The ROS gmapping package

TABLE 1. Parameter value.

Parameter	Value	Parameter	Value
α	1	β	1
γ	0.5	ε	0.1
φ	0.1	ϕ	0.2
μ	0.3	σ	0.3
ρ	1	r	5m
m	60	s	5m
$\eta_{(10*10)}$	1m	$\eta_{(20*20)}$	2m
$\eta_{(40*40)}$	4m	$\eta_{(60*60)}$	6m
$s_{fixed(10*10)}$	1m	$s_{fixed(20*20)}$	2m
$s_{fixed(40*40)}$	4m	$s_{fixed(60*60)}$	6m

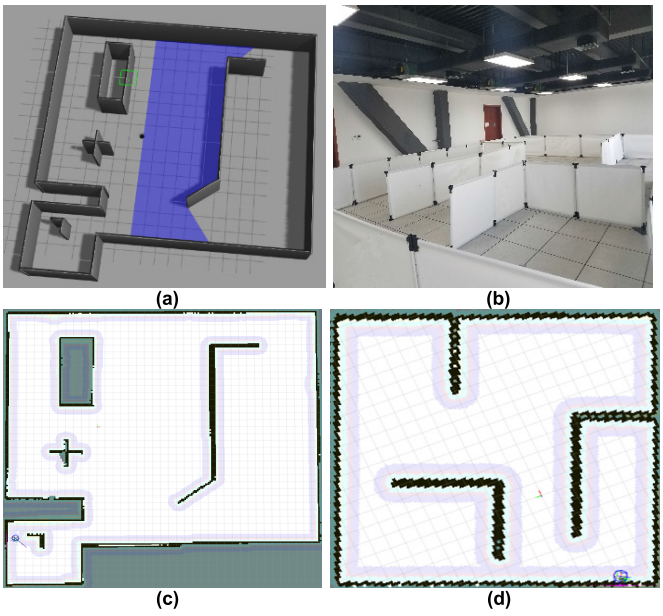


FIGURE 4. Experiment environment. (a) Simulation environments; (b) real environments; (c) occupancy grid generated for the simulation environments; (d) occupancy grid generated for the real environments.

is used for generating the map and localizing the robot in our experiments. And the ROS Navigation stack is used to control and direct the robot towards exploration goals. In addition, the system parameters are shown in Table 1.

For the simulation environment, we use Gazebo simulator [34] to build a closed space, which contains some rooms and obstacles, as shown in Fig.4 (a) below. Considering the influence of the changes in map size, different map sizes (20*20m, 40*40m, 60*60m) are used. The robot's radius is 0.2m and the laser sensor's range is set to 10 m. Fig.4 (c) represents the 2D occupancy grid map built in simulated environment with the size of 20*20m.

In the real environment, we use the baffle to build a space of 10m*10m, as shown in Fig.4 (b) below. The mobile robot platform used in the experiments is the EAI-BOT Dashgo-D1. It is equipped with a Hokuyo UST-10LX 2D laser sensor (10m detection range and 270° filed of views). Fig.4 (d) is the 2D occupancy grid map built in real environment.

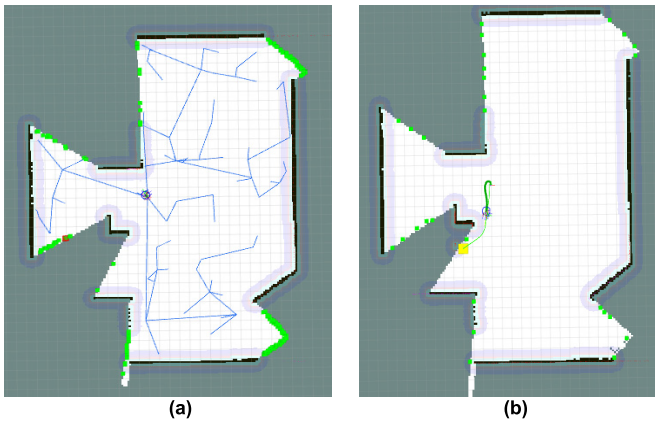


FIGURE 5. (a) Frontier points generated before optimization; (b) frontier points generated after optimization.

B. THE RESULT OF FRONTIER POINTS OPTIMIZATION

Before the autonomous robotic exploration, let the robot take a turn first to get a local map. In the local map, we generate frontier points using the RRT algorithm, as shown in the following Fig.5 (a). Due to the randomness of RRT algorithm, the positions of the frontier points are all random. It can be seen that some frontiers have a lot of frontier points, some frontiers only have few frontier points. Moreover, on each frontier of the map, the distribution of frontier points is also uneven. Fig.5 (b) is the frontier points generated using our proposed algorithm. As shown in the Fig.5 (b), after the optimization, the number of frontier points has been greatly reduced, and the frontier points on each frontier are basically uniformly distributed. The yellow point is the optimal frontier point in the current situation which is calculated according to the frontier points evaluation function we defined.

C. THE RESULT OF MULTISTEP EXPLORATION STRATEGY

In this subsection, we compare the effects of different path planning strategies. As shown in Fig.6 (a) and Fig.6 (b), the robot directly plans the path from the current position to the target frontier point and search for the next target frontier point until the previous target frontier is reached under the traditional global path planning strategy. This strategy will lead to a problem: new frontier points will be generated during the exploration process, and the position of these frontier points may be better than the current target frontier point. The situation that robot chooses new frontier point to explore after it has reached the previous target frontier point may cause the repetition of robot's exploration path, making the exploration distance to increase. However, in the multistep path planning strategy, every time the movement distance of the robot reaches the local exploration path step size s_{fixed}, we recalculate and reselect the optimal frontier point to prevent this kind of situation from happening. It can be seen from the following Fig.6 (c) that before the robot reaches the target frontier point in Fig.6 (a), the current optimal frontier point has changed. Therefore, the robot has planned a new path.

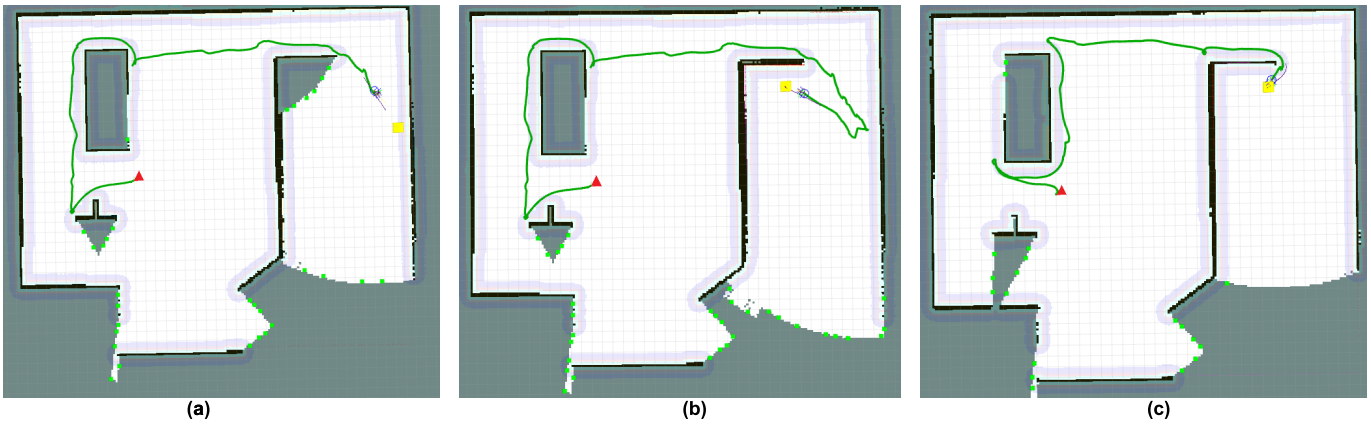


FIGURE 6. The result of multistep exploration strategy, the red triangle represents the starting point, the yellow point represents target frontier point. (a) Move to the target frontier point; (b) move to the next target frontier point; (c) move to the target frontier of the multistep path planning.

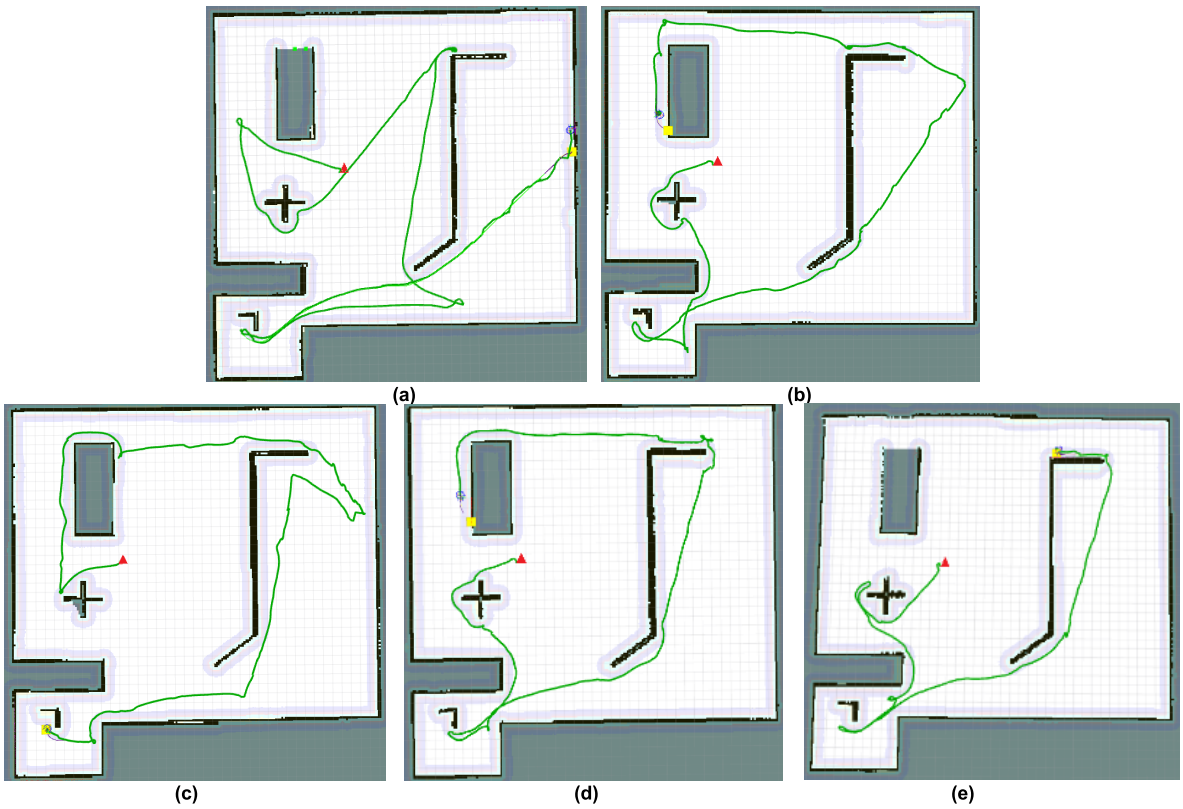


FIGURE 7. The exploration trajectory of each strategy in the simulation map of 40*40 m. (a) The exploration trajectory of RANDOM strategy; (b) the exploration trajectory of NEAREST strategy; (c) the exploration trajectory of GREEDY strategy; (d) the exploration trajectory of UMARI strategy; (e) the exploration trajectory of RFPO strategy.

The result is that the path length in Fig.6 (c) is obviously shorter than the Fig.6 (a) and Fig.6 (b).

D. COMPARSION WITH OTHER STRATEGIES

We totally perform 200 sets of experiments to compare the strategy we proposed with other four strategies. The idea of strategy 1 is to select the frontier point randomly for exploration which we call it RANDOM. The idea of Strategy 2 is to select the nearest frontier point to the robot and we use NEAREST to represent it. Strategy 3 uses the idea of greedy [35] algorithm, so we denote it as GREEDY in this paper. Strategy 4 is proposed by Umari and Mukhopadhyay [8] and we use UMARI to describe it. The strategy which is proposed by us in this paper is called RFPO. In order to compare the impact of different map sizes on exploration strategies, we use 4 different sizes of map for experiments (a real map and three different sizes of simulation map). For each map, 50 sets of exploration runs are carried out. These 50 exploration runs are divided into 5 sets and each set represents an exploration strategy. In the simulation map of 40*40 m, for each strategy, we select an experiment result from the 10 exploration runs to show the robot's exploration trajectories. The results are shown as the Fig.7.

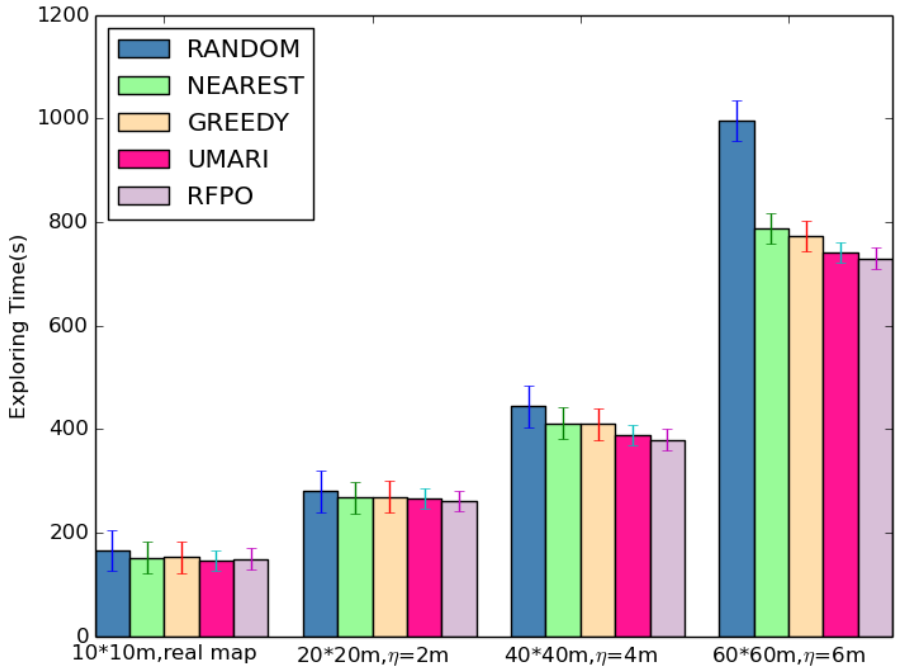


FIGURE 8. The figure of exploration time when the exploration has finished.

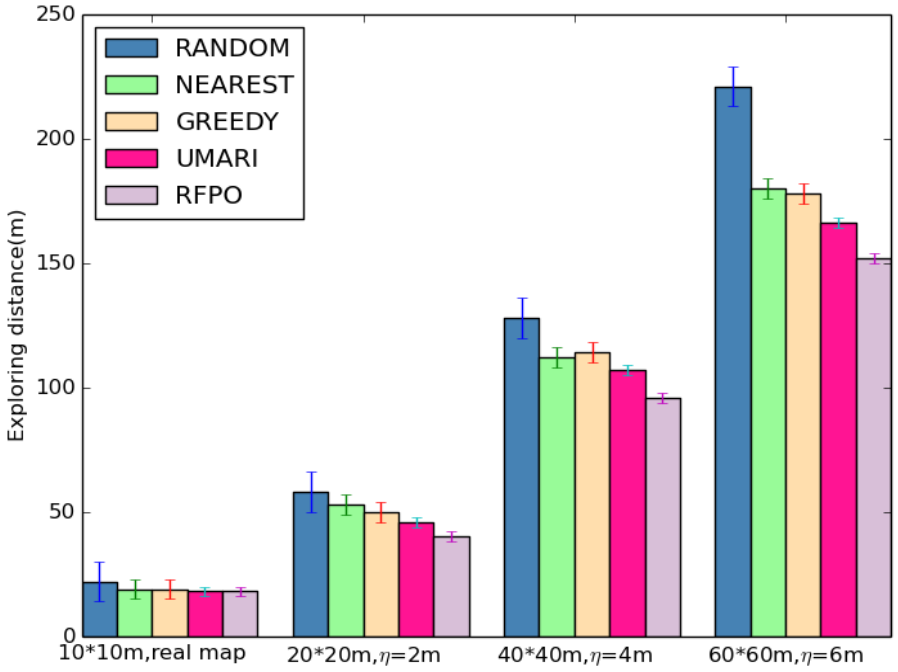


FIGURE 9. The figure of exploration distance when the exploration has been finished.

We made statistics on the exploration time and exploration distance using the data of the 200 sets of experiments, and plotted the following figures. Fig.8 shows the exploration time when the exploration has been finished using different exploration strategies in the four different maps, and Fig.9 shows the exploration distance when the exploration has been finished using different exploration strategies in the four different maps. In order to reduce the effect of the random tree's growth rate on the experimental results, we set the corresponding growth rate according to different map sizes. From the figures we can see that, the larger size of the map is, the more obvious of the difference in exploration efficiency between different strategies are. In the simulation map of 60*60 m, the average exploration time of our proposed strategy respectively decreased by 26.71%, 7.36%, 5.56%, 1.62% compared with the other four strategies; the average exploration distance respectively decreased by 31.22%, 15.56%, 14.61%, 8.43%. For the RANDOM strategy, since

the target frontier point is randomly selected each time, the robot will take a lot of repeated routes, so the exploration time and the exploration distance will increase. The NEAREST strategy and GREEDY strategy may cause exploration into a local optimal problem and affect the efficiency of exploration. Strategy of UMARI directly plans the path from the robot's current location to the exploration target point, which may lead to problems in Fig.6. In our exploration strategy, we optimize the generated random frontier points and select the point with the largest evaluation value as the target frontier point in the optimization results. In the path planning section, the multistep exploration strategy is implemented to reduce the probability of the robot's exploration path repetition. The experimental results show that, whether compared with the exploration time or the exploration distance, the effect of the exploration strategy proposed by us performs better than other strategies, which prove the effectiveness of our strategy.

V. CONCLUSION

In this paper, an autonomous exploration strategy of robot based on frontier point optimization and multistep path planning is presented. This strategy can drive robot to explore unknown environments and build corresponding 2D occupancy grid maps with a high efficiency without human intervention. In this exploration strategy, we use the RRT algorithm to generate the frontier points and propose the RFPO algorithm to optimize these frontier points. The frontier points evaluation function is defined to select the current optimal frontier point for exploration. In the path planning section, we set a local exploration path step size, and reselect the target frontier point for exploration when the movement distance of robot reaches the local exploration path step size so as to reduce the possibility that the robot may take some repetitive paths. We carry out relevant experiments in the simulation environments and real environment. The experimental results verify the effectiveness of our proposed strategy. Since we currently only use odometer data combining laser sensor data to build the 2D occupancy grid map, the information contained in the map is relatively little. Next, we intend to fuse the visual sensor data into the autonomous exploration. The advantage of visual sensor data is that it can obtain more environmental information. These data can be fused to build a map with richer information for the navigation task and other related work later. In addition, how to coordinate the exploration of multiple robots [36] efficiently is also an important task that we are prepared to do.

REFERENCES

[1] V. Indelman, L. Carlone, and F. Dellaert, "Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 849–882, 2015.

[2] D. P. Ström, F. Nenci, and C. Stachniss, "Predictive exploration considering previously mapped environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 2761–2766.

[3] P. G. C. N. Senarathne and D. Wang, "Incremental algorithms for Safe and Reachable Frontier Detection for robot exploration," *Robot. Auton. Syst.*, vol. 72, pp. 189–206, Oct. 2015.

[4] M. Keidar and G. A. Kaminka, "Efficient frontier detection for robot exploration," *Int. J. Robot. Res.*, vol. 33, no. 2, pp. 215–236, 2014.

[5] M. Keidar and G. A. Kaminka, "Robot exploration with fast frontier detection: Theory and experiments," in *Proc. Int. Conf. Auton. Agents Multiagent Syst.*, 2013, pp. 113–120.

[6] P. G. C. N. Senarathne, D. Wang, Z. Wang, and Q. Chen, "Efficient frontier detection and management for robot exploration," in *Proc. IEEE Int. Conf. Cyber Technol. Automat., Control Intell. Syst.*, May 2013, pp. 114–119.

[7] A. Gautam, B. Jha, G. Kumar, J. K. Murthy, S. A. Ram, and S. Mohan, "FAST: Synchronous frontier allocation for scalable online multi-robot terrain coverage," *J. Intell. Robotic Syst.*, vol. 87, no. 3, pp. 545–564, 2017.

[8] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 1396–1402.

[9] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat., (Cira)*, Jul. 1997, pp. 146–151.

[10] R. G. Simmons *et al.*, "Coordination for multi-robot exploration and mapping," in *Proc. 17th Nat. Conf. Artif. Intell. 12th Conf. Innov. Appl. Artif. Intell.*, 2000, pp. 852–858.

[11] S. J. Moorehead, R. Simmons, and W. L. Whittaker, "Autonomous exploration using multiple sources of information," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 3, May 2001, pp. 3098–3103.

[12] L. Carlone and D. Lyons, "Uncertainty-constrained robot exploration: A mixed-integer linear programming approach," in *Proc. IEEE Int. Conf. Robot. Autom.*, May/Jun. 2014, pp. 1140–1147.

[13] Y. Mei, Y.-H. Lu, C. S. G. Lee, and Y. C. Hu, "Energy-efficient mobile robot exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2006, pp. 505–511.

[14] D. P. Ström, I. Bogoslavskyi, and C. Stachniss, "Robust exploration and homing for autonomous robots," *Robot. Auton. Syst.*, vol. 90, pp. 125–135, Apr. 2017.

[15] A. Gautam, J. K. Murthy, G. Kumar, S. P. A. Ram, B. Jha, and S. Mohan, "Cluster, allocate, cover: An efficient approach for multi-robot coverage," in *Proc. IEEE Int. Conf. Syst.*, Oct. 2016, pp. 197–203.

[16] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon 'next-best-view' planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 1462–1468.

[17] E. Masehian and H. Kakahaji, "NRR: A nonholonomic random replanner for navigation of car-like robots in unknown environments," *Robotica*, vol. 32, no. 7, pp. 1101–1123, 2014.

[18] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robot. Auto. Syst.*, vol. 83, pp. 15–31, Sep. 2016.

[19] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," *Robot., Sci. Syst.*, vol. 2, pp. 65–72, Jun. 2005.

[20] M. Elhoseny, A. Shehab, and X. Yuan, "Optimizing robot path in dynamic environments using genetic algorithm and Bezier curve," *J. Intell. Fuzzy Syst.*, vol. 33, no. 4, pp. 2305–2316, 2017.

[21] P. G. C. N. Senarathne, "Frontier based exploration with task cancellation," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot.*, Oct. 2014, pp. 1–6.

[22] H. H. González-Baños and J. C. Latombe, "Navigation strategies for exploring indoor environments," *Int. J. Robot. Res.*, vol. 21, no. 10, pp. 829–848, 2001.

[23] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, Sep./Oct. 2002, pp. 534–539.

[24] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, Sep./Oct. 2002, pp. 540–545

[25] G. Grisettiyz, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 2432–2437.

[26] Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.

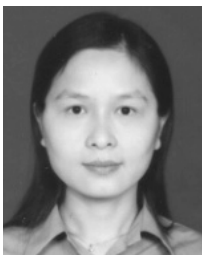
[27] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., vol. 98, no. 11, Oct. 1998.

[28] K. N. Krishnanand and D. Ghose, "Detection of multiple source locations using a glowworm metaphor with applications to collective robotics," in *Proc. IEEE Swarm Intell. Symp. (SIS)*, Jun. 2005, pp. 84–91.

[29] K. N. Krishnanand and D. Ghose, "Multimodal Function Optimization using a Glowworm Metaphor with Applications to Collective Robotics," in *Proc. Indian Int. Conf. Artif. Intell.*, Dec. 2005, pp. 328–346.
[30] Internet. *Roulette*. Accessed: Mar. 8, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Roulette
[31] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997.
[32] M. Seder and I. Petrovic, "Dynamic window based approach to mobile robot motion control in the presence of moving obstacles," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 1986–1991.
[33] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009, pp. 1–6.
[34] Internet. *Gazebo simulator*. Accessed: Mar. 8, 2019. [Online]. Available: http://gazebosim.org/
[35] F. Amigoni and A. Gallo, "A multi-objective exploration strategy for mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2005, pp. 3850–3855.
[36] A. Q. Li, R. Cipolleschi, M. Giusto, and F. Amigoni, "A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings," *Auton. Robots*, vol. 40, no. 4, pp. 581–597, 2016.

BAOFU FANG received the Ph.D. degree in computer application technology from the Harbin Institute of Technology (HIT), China, in 2013. He joined the Department of Computer Science and Technology, School of Computer and Information, Hefei University of Technology, as an Associate Professor, in 2010, and a Master's Supervisor, in 2011. He is the Technology Chair of the Anhui Robot Competition, a member of the Standing Committee of the China Association of Artificial Intelligence (CAAI) Young Committee and the Standing Committee of the China Association of Artificial Intelligence (CAAI) Robot and Culture Committee. He have many funds sponsored by the National High Technology Research and Development Program of China (863), the Natural Science Foundation of Anhui Province, the Fundamental Research Funds for the Central Universities, the Hefei University of Technology, and some enterprise and company. He has been publishing about 60 more research papers in journal and conferences, since 2002.

JIANFENG DING received the B.S. degree in computer science and technology from Northwest University, China, in 2016. He is currently pursuing the M.E. degree in computer science and technology with the School of Computer and Information, Hefei University of Technology, Hefei, China. His research interest includes SLAM.



ZAIJUN WANG received the M.Eng. degree in materials science and engineering from Xihua University, China, in 2008. She is currently an Associate Professor with the CAAC Academy of Flight Technology and Safety, Civil Aviation Flight University of China. Her research interests include multirobot task allocation, artificial intelligence, and data mining.