Search Planning and Analysis for Mobile Targets with Robots

Shujin Ye(✉), Wai Kit Wong, and Hai Liu

Department of Computing, The Hang Seng University of Hong Kong, Siu Lek Yuen, Hong Kong
{sye,wongwk,hliu}@hsu.edu.hk

Abstract. With robotics technologies advancing rapidly, there are many new robotics applications such as surveillance, mining tasks, search and rescue, and autonomous armies. In this work, we focus on use of robots for target searching. For example, a collection of Unmanned Aerial Vehicle (UAV) could be sent to search for survivor targets in disaster rescue missions. We assume that there are multiple targets. The moving speeds and directions of the targets are unknown. Our objective is to minimize the searching latency which is critical in search and rescue applications. Our basic idea is to partition the search area into grid cells and apply the divide-and-conquer approach. We propose two searching strategies, namely, the circuit strategy and the rebound strategy. The robots search the cells in a Hamiltonian circuit in the circuit strategy while they backtrack in the rebound strategy. We prove that the expected searching latency of the circuit strategy for a moving target is upper bounded by $\frac{3n^2-4n+3}{2n}$ where n is the number of grid cells of the search region. In case of a static or suerfast target, we derive the expected searching latency of the two strategies. Simulations are conducted and the results show that the circuit strategy outperforms the rebound strategy.

Keywords: Robot search · Mobile target · Search planning and analysis

1 Introduction

From small toasters to huge industrial machines, robots are already an indispensable part of our daily lives. Automatic robots such as Unmanned Aerial Vehicle (UAV) can now be bought in many shops and are easily affordable by individuals. Some UAVs can be bought at several hundreds US dollars. It is not surprise to see UAVs flying nowadays. Apart from aerial robots, there are also other kinds of robots such as unmanned vehicles (e.g., Google driverless cars), AUVs/UUVs (autonomous underwater vehicle/unmanned underwater vehicles), and unmanned ships. The robots carry sensors such as accelerometers, infrared detectors, microphones and cameras. They can be used in many applications

https://doi.org/10.1007/978-3-030-38819-5_1

including surveillance, search and rescue, payload delivery and military missions [1,2].

In this paper, we focus on searching missions using robots [7,25]. Figure 1 shows an example of the searching mission, where several robots are deployed to search for multiple targets in an area. In July 2018, a 44-year-old paraglider was blown off course from Sunset Peak in Lantau South Country Park in Hong Kong[1]. The missing person was believed to have fallen into the nearby jungle or the sea. It is difficult and slow to deploy ground machines or human forces to search in the jungle or the sea. The Hong Kong police considered searching for the missing person using UAVs, which could be massively deployed to assist the search at land, sea, and air at low visibilities. There could be multiple targets to be rescued and they may move from time to time. For example, in the incident of MH370[2], there are over 200 people missing and they may move following the ocean current. Using a large number of robots to assist in the search can increase the chance of locating the targets in the huge search area and hence may save more lives.

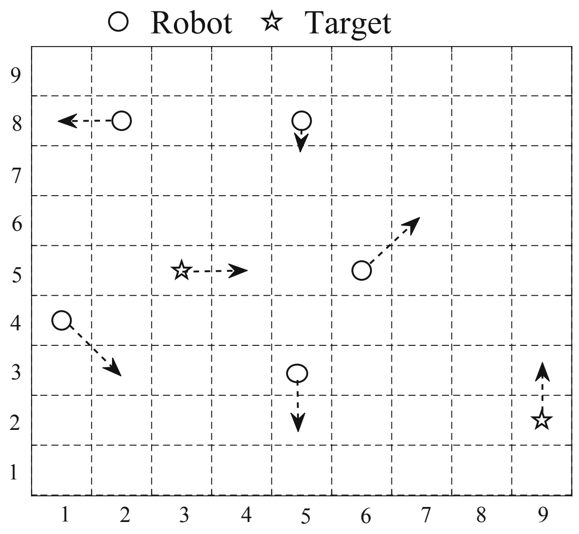


Fig. 1. Robots searching for mobile targets in the search area

Although searching for mobile targets using robots is common in practice, this topic has received little attention. To our best knowledge, we are the first to study and analyze the strategies of robot-based mobile target searching. We assume no information, such as speed and direction, about the target's movement

[1] https://www.hongkongfp.com/2018/07/27/hong-kong-paraglider-missing-since-sunday-found-dead-lantau-island/.

[2] https://www.usatoday.com/story/news/world/2014/03/07/malaysia-airlines-beijing-flight-missing/6187779/.

is known. We use a divide and conquer approach to let the robots search in different areas. Each robot searches its assigned area in a pre-set path. There are two strategies: (1) Circuit strategy and (2) Rebound strategy. The search strategies are divided into two phases. In the first phase, both strategies detect every place in the map once to find the targets. However, as targets may move, the robots may miss the targets in the first phase. Then, in the second phase, the circuit strategy repeats the same path in the first phase until the robot runs out of energy. In the rebound strategy, a robot backtracks (i.e., going back along the path that it followed in the first phase) until its energy is depleted. Interestingly, as we have proven in this paper, the circuit strategy has a lower *searching latency* than the rebound strategy, i.e., the circuit strategy can find the targets in a shorter time. We have also analyzed the expected searching latency of our strategies. In specific, with n being the number of cells in the search area, we proved the following:

1. The expected searching latency of the circuit strategy is upper bounded by $\frac{3n^2-4n+3}{2n}$.
2. The expected searching latency of both strategies is $\frac{n+1}{2}$ when the target is static.
3. The expected searching latency of any strategy is n when the target is super-fast and can move to any location in the search area at any time.

The simulation results agree with our theoretical results and show that the circuit strategy has the lowest searching latency.

2 Related Works

Motion control of robots was modeled in [3–5,10,22,28] and has been applied in a number of applications, such as target searching [8,18,19] target tracking [16,23,24], formation of specific topologies and patterns [12,15,26], coverage area maximization [6,14,21,27], vehicle behavior description [11,13], and network connection maintenance [9,17,19]. We focus on the target searching literature as follows.

Several studies have examined searches for a single static target. Dimidov et al. [8] analyzed the efficiency of two classes of random walk search strategies, namely the correlated random walk and Lévy walk, for discovering a static target in either a bounded or open space. The search strategies were tested through simulations that used up to 30 real robots (Kilobots). During the experiment, communication between robots included sharing information about discovery of the target. Sakthivelmurugan et al. [20] introduced a number of searching strategies for the detection and retrieval of a static target, including the straight-line, parallel-line, divider, expanding square, and parallel sweep approaches. Experiments were conducted with up to four robots in an environment with known boundaries and showed that a parallel sweep with the divider approach was the most efficient strategy.

Searching for multiple targets was studied by Rango et al. [18]. They considered the mine detection problem using a modified ant colony algorithm. Given

Table 1. A summary of differences between our work and current literature

	Single/multiple targets	Static/mobile targets
This paper	Multiple targets	Both static and mobile targets
[19]	Single target	Static target
[8]	Single target	Static target
[18]	Multiple targets	Static targets

multiple mines that were randomly distributed in an unexplored area, the problem is to use robots to explore the area and detect all of the mines in the minimum time. During the searching process, the robots lay repelling anti-pheromones on the explored area. When a robot decides for its next next movement, it perceives pheromone information from their surroundings and travels to undetected regions with the least pheromone intensity. Once one or more robots discovered a mine, attractive pheromones were deposited to recruit other robots. After the required number of robots were attracted to the mine location, they worked cooperatively to disarm the mine.

Although the robot-based target searching has been studied, the previous works focused on the static target. To the best of our knowledge, this paper is the first one to consider mobile target searching using robots. In Table 1, we summarize the differences between our work and current literature.

3 Problem Definition and System Models

There are M targets (e.g., missing persons). They are located within a rectangular *search area* of size $w \times \ell$. They may move from time to time. The speed and direction of the targets are unknown and they can change at any time. Targets are assumed not to move outside the search area. There are N homogeneous robots (e.g., UAVs). Each robot is given a unique integer 1 to N as its ID. Each robot has a lifetime of L, representing the time it can spend to search for the target, e.g., before the battery is empty. Each robot has sensing ability so that it can detect a target within a small range. We define the *searching latency* of i-th target, denoted as T_i, as the time to locate the target since the start of the search. The problem is to design searching plans of the robots so that the chance for the robots to find all targets is maximized and the average searching time $(\frac{\sum_{i=1}^{M} T_i}{M})$ is minimized.

First, we discretize the problem as following. The sensing range of each robot is assumed to be a square[3] of length k so that the robot can detect the target if it is located in the square. We divide the search area into $w_c \times \ell_c$ cells where $w_c = \lceil \frac{w}{k} \rceil$ and $b = \lceil \frac{\ell}{k} \rceil$. We write (x, y) as the coordinate of each cell. $(1, 1)$ represents the bottom left corner while (w_c, ℓ_c) represents the upper right corner. Let $n = w_c \times \ell_c$ be the total number of cells of the search area. In our system,

[3] The actual sensing range could be a circle containing the square.

Table 2. Important notations.

Notation	Definition
w_c	Number of cells in width of the search area
ℓ_c	Number of cells in length of the search area
n	Total number of cells of the search area, $n = w_c \times \ell_c$
N	Number of robots
M	Number of targets
L	Lifetime of the robots
T_i	Searching latency of i-th target
t	The current timeslot

we consider that n is significantly larger than number of robots N as the search area in reality is normally very large. Time is also discretized (time-slotted). We define 1 unit of timeslot as the time taken by the robot to move from a cell to a neighboring cell (including diagonally) and perform the detection. In others words, after a robot has finished the detection at a cell, it can move in 8 directions and reach the next cell to perform the detection. Each robot performs the movement and detection for exactly 1 cell in 1 timeslot, i.e., the speed of the robot is 1 (cell per timeslot). A target is always located in one of the n cells of the search area. If a target and a robot are in the same cell at the same timeslot, the target is detected. The problem now becomes to design search paths of robots on the cells. Table 2 summarizes the major notations used in the paper.

4 Search Strategy of Robots

We use a two-phase search strategy as described below.

Phase 1: Explore all Cells Once. Without any information about the target's location, the first goal in the searching task is to check all the cells in the search area once. We use a divide and conquer approach to let the robots search in different regions at the same time. We take a side (width or length) of the search area, and divide it into N strips along this side, as shown in Fig. 2. Each robot is assigned a strip to search for targets and it visits every cell in the strip to find the targets. The side is selected in order to minimize the difference between the areas of the strips. Say for example, if w_c is divisible by N, we divide the search area along the width. Each strip is an equi-width rectangle of size $\frac{w_c}{N} \times \ell$. In general, say we divide the search area along the width and let $r = w_c \mod N$. For robots with ID $\leq r$, each is assigned a strip of width $\lceil \frac{w_c}{N} \rceil$ and length ℓ_c. For robots with ID $> r$, each is assigned a strip of width $\lfloor \frac{w_c}{N} \rfloor$ and length ℓ_c. Each robot starts its search at the bottom left corner of its assigned strip at $t = 1$.

Phase 2: Re-visit Until All Targets are Found. If the targets are static, i.e., they do not move, it is guaranteed to find all targets in phase 1. Since the targets may move, the robots may miss some targets. The robots repeat the search until

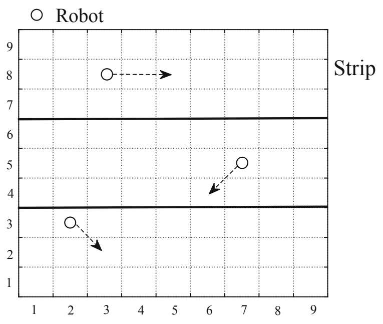
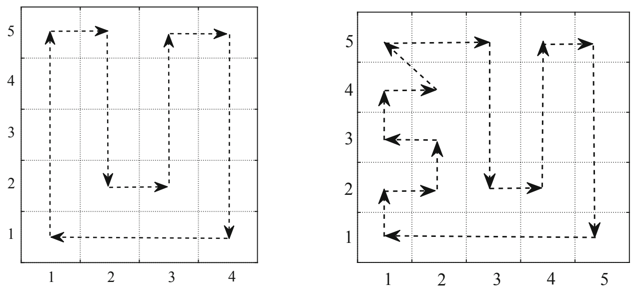


Fig. 2. Searching area is divided into two strips for two robots



(a) One side has even number of cells

(b) Both sides have odd number of cells

Fig. 3. Search paths of the circuit strategy for different dimensions of the search area.

all targets are found. We consider the following two searching strategies for each robot.

Circuit Strategy. Each robot travels in a *Hamiltonian circuit* to visit all the cells in the assigned strip. After visiting all the cells once, the robot ends at the staring location. It repeats the circuit again and again until all targets are found. Hamiltonian circuit always exist when the width and length of the strip are over 1. Figure 3 illustrates the search paths of the circuit strategy.

Rebound Strategy. Each robot travels in a *Hamiltonian path*. After it visits all the cells once, it starts at the ending point and travels back the same Hamiltonian path in the opposite direction. It repeats the path back and forth until all targets are found. Hamiltonian path always exist. Figure 4 illustrates the search path of the rebound strategy.

5 Analysis on Searching Latency

We analyze the searching latency of the proposed searching strategies theoretically. We assume there is one robot only in the search area. It is trivial to extend

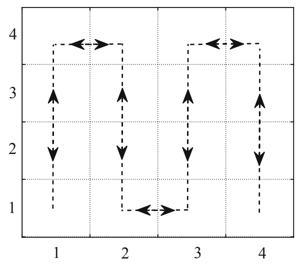


Fig. 4. Search path of the rebound strategy.

to the multi-robot case as each robot searches in a non-overlapping area. If the targets are all independent, the expected searching latency of any of them is the same. Our analysis focuses on the searching latency of j-th target, T_j. We divide the analysis into three parts:

Static Target. The target does not move but its location is unknown.
Superfast Target. We consider the target's speed v is over the size of the search map, i.e., $v \geq \max(w_c, \ell_c)$. The target can move from any cell to any cell between any two timeslots. In this case, the locations of the target at two consecutive timeslots are independent.
Mobile Target. The target may move and its speed v is smaller than the size of the search map, i.e., $v < \max(w_c, \ell_c)$.

5.1 Analysis on Static Target

As the robot searches in all cells, it is guaranteed to find the target. In Theorem 1, we show that both strategies have the same expected searching latency.

Theorem 1. *If the target is static, then the expected searching latency of a robot using either circuit strategy or rebound strategy is $E(T_j) = \frac{n+1}{2}$, where n is the number of cells of the search area.*

Proof. Since there are n cells, the target must be found at or before $t = n$. There is a probability of $\frac{1}{n}$ to find the target at $t = i$ for $i = 1$ to n. We have

$$\begin{aligned} E(T_j) &= \sum_{i=1}^{n} \Pr(\text{Target found at } t = i) \cdot i \\ &= \frac{1}{n} \sum_{i=1}^{n} i \\ &= \frac{1}{n} \cdot \frac{n(n+1)}{2} \\ &= \frac{n+1}{2} \end{aligned}$$

□

5.2 Analysis on Superfast Target

In Theorem 2, we show that any strategy has the same expected searching latency.

Theorem 2. *If the target is mobile with speed $\geq \max(w_c, \ell_c)$, then the expected searching latency of a robot using any strategy is $E(T_j) = n$, where n is the number of cells of the search area.*

Proof. The probability of the target appearing in any cell at any time $t = i$ is the same, i.e., $\Pr(\text{Robot meets target at } t = i) = \frac{1}{n}$. Note that the above probability is the same regardless of which strategy is used. The expected latency is

$$E(T_j) = \frac{1}{\Pr(\text{Robot meets target at } t = i)} = n$$

□

5.3 Analysis on Mobile Target

Every time a robot explores a cell and cannot find the target there, we can conclude that the target must be in one of the remaining cells. We model the target's movement as a probabilistic model where the target may stay at the same cell in the next timeslot with a probability p or may move to a neighboring cell with even probability. We can estimate the probability of finding the target at different cells using the following.

Let x be a cell in the search area and $\mathbb{N}_x$ be the set of reachable cells of x. Let $P_i^{(x)}$ be the probability of the target being located at cell x at $t = i$. Assume the robot does not explore x at $t = i + 1$. We have

$$\begin{aligned} P_{i+1}^{(x)} =& P_i^{(x)} \times \Pr(\text{the target stays at}\, x) + \sum_{y \in \mathbb{N}_x} P_i^{(y)} \times \Pr(\text{the targets leaves}\, y) \\ & \times \Pr(\text{the target moves to}\, x \mid \text{the targets leaves}\, y) \\ =& pP_i^{(x)} + (1-p) \sum_{y \in \mathbb{N}_x} \frac{P_i^{(y)}}{|\mathbb{N}_y|} \end{aligned} \tag{1}$$

At $t = i + 1$, the robot explores one of the cells say z. Suppose the target is not found at z. Let $\widehat{P_{i+1}^{(x)}}$ be the probability of the target being located at cell x at $t = i + 1$ given that the target is not found at z. We have $\widehat{P_{i+1}^{(z)}} = 0$ and

$$\widehat{P_{i+1}^{(x)}} = \frac{P_{i+1}^{(x)}}{1 - P_{i+1}^{(z)}} \tag{2}$$

We use the special case where $w_c = \ell_c = 2$ to illustrate the idea, i.e., the search area has four cells in total. $|\mathbb{N}_x| = 3$ for all cells x. Suppose $p = 0.5$.

Figure 5 illustrates how the probabilities of finding the target at different cells change over time. Initially, at $t = 1$, each of the three unexplored cells has the same probability ($\frac{1}{3}$) to find the target. At $t = 2$, we calculate the probabilities according to Eqs. 1 and 2. $P_2^{(1,1)} = 0.5(\frac{1}{3}) + 0.5(\frac{\frac{1}{3}+\frac{1}{3}+0}{3}) = \frac{5}{18} = 0.278$. Similarly, $P_2^{(2,1)} = \frac{5}{18} = 0.278$. Since the robot explored (1, 1) and the target is not found, $\widehat{P_2^{(1,1)}} = 0$ (lower left cell). We have $\widehat{P_2^{(2,1)}} = \frac{\frac{5}{18}}{1-\frac{5}{18}} = \frac{5}{13} = 0.385$ (lower right cell).

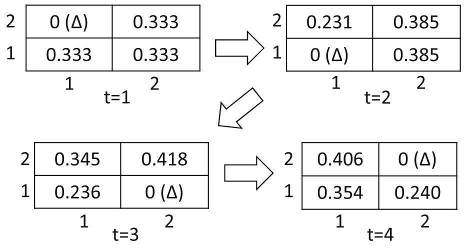


Fig. 5. Illustration of the probability of finding the target at different locations in a 2×2 search area from $t = 1$ to 4, given the target is not found. $p = 0.5$. Δ denotes the location explored by the robot at that timeslot.

Observe that *the probability to find the target is not even at all the cells.* A cell that is explored recently has the lowest probability to find the target. This probability increases as time goes until it is explored again. The same can be observed in the general case. From Eq. 1, $P_{i+1}^{(x)}$ is the smallest when the robot has explored x at $t = i$, i.e., $P_i^{(x)} = 0$. Assume $|\mathbb{N}_x| = |\mathbb{N}_y|$ for any cells x and y. The change in $P_i^{(x)}$, denoted as $\Delta P_i^{(x)}$, can be calculated as

$$\Delta P_i^{(x)} = P_{i+1}^{(x)} - P_i^{(x)} = (1-p)(\overline{P_i^{(y)}} - P_i^{(x)}) \tag{3}$$

where $\overline{P_i^{(y)}} = \frac{\sum_{y \in \mathbb{N}_x} P_i^{(y)}}{|\mathbb{N}_x|}$ denotes the average of $P_i^{(y)}$ for $y \in \mathbb{N}_x$. When $P_i^{(x)}$ is small, $\Delta P_i^{(x)}$ is large. As $P_i^{(x)}$ gets larger, $\Delta P_i^{(x)}$ decreases, until $P_i^{(x)}$ approaches the average of $P_i^{(y)}$ of the neighboring cells.

Summary. To maximize the chance to find the target, it is preferred to explore the cells that are not explored for the longest time at each timeslot. In the first phase of the search, both the circuit strategy and the rebound strategy visit the unexplored cells once. This matches the principle above. At the beginning of the second phase, the circuit strategy and the rebound strategy restart the search at different routes. The circuit strategy explores the cells that have not been detected for a longer time. The rebound strategy, in contrast, visits the cells that are just visited not long ago. The circuit strategy is expected to have a better chance to find the target during this period. The rebound strategy leaves the cells around the starting point of the search path to be explored later. This increases the time gap between two detections on the same cell. However, as we have discussed in Eq. 3, the gain in the probability to find the target is marginal. Thus, we expect the circuit strategy to outperform rebound strategy.

Upper Bound of Expected Latency of the Circuit Strategy. The circuit strategy and the rebound strategy are simple and intuitive. They are similar in the sense that they travel through all the cells repeatedly to find the targets. However, as we have showed, circuit strategy has a smaller expected searching latency than rebound strategy. Circuit strategy is suggested in practice. In Theorem 3, we provide an upper bound of the expected searching latency of the circuit strategy.

Theorem 3. *The expected searching latency of the circuit strategy $E(T_j)$ is upper bounded by $\frac{3n^2-4n+3}{2n}$ searching a mobile target in a search area of n cells.*

Proof. The search path of the robot is static. Say, the path in Fig. 3(b) is used. Suppose now the target is at $(1,1)$ at $t=i$ and the robot is not here. We know that the robot cannot be at $(1,2)$ at $t=i+1$. The probability of being found at $t=i+1$ if the target moves to (1, 2) is 0. We call this location the *blind spot*. The robot may appear in any of the remaining $n-1$ cells at $t=i+1$. Let C be the set of candidate locations that the target may move to at $t=i+1$. The target moves to any one of the locations in C with even probability. Let p be the probability of the target being found at $t=i+1$. If the blind spot is in C, we have $p=\sum_{i=1}^{|C|-1}\frac{1}{|C|}\cdot\frac{1}{n-1}=\frac{m-1}{m(n-1)}$. p is the smallest when $|C|$ is the smallest with the blind spot being one of the candidates in C. This happens when the target is moving at the speed of 1 cell per timeslot and is located at the corner, say $(1,1)$. There are only 3 candidate locations in this scenario: $(1,2)$ (blind spot), $(2,1)$, $(2,2)$. We denote the smallest of p, $p_{min}=\frac{2}{3(n-1)}$.

$E(T_j)$ is the largest (the upper bound) when the probability to find the target at any time is p_{min}. We denote it as $E_{max}(T_j)$. We calculate $E_{max}(T_j)$ as following.

Let E_i be the event of the target being found at $t=i$ and $\overline{S_i}$ be $\overline{E_1}\wedge\overline{E_2}\wedge\ldots\wedge\overline{E_i}$. At $t=1$, the target is located at a random position. $\Pr(E_1)=\frac{1}{n}$. At $t=2$, as we discussed above, $\Pr(E_i|S_{i-1})=p_{min}$ for $i>1$. We have $\Pr(S_i)=\frac{n-1}{n}\cdot(1-p_{min})^{i-1}$ for $i\geq 1$.

$$\begin{aligned}
E_{max}(T_j) &= \Pr(E_1)\cdot 1+\sum_{i=2}^{\infty}\Pr(S_{i-1})\cdot\Pr(E_i|S_{i-1})\cdot i\\
&= \frac{1}{n}+\sum_{i=2}^{\infty}\frac{n-1}{n}\cdot(\frac{3n-5}{3n-3})^{i-2}\cdot\frac{2}{3(n-1)}\cdot i\\
&= \frac{1}{n}+\frac{2}{3n}\sum_{i=2}^{\infty}(\frac{3n-5}{3n-3})^{i-2}\cdot i
\end{aligned}$$

The term $X = \sum_{i=2}^{\infty} (\frac{3n-5}{3n-3})^{i-2} \cdot i$ is an arithmetico-geometric sequence. Let $q = \frac{3n-5}{3n-3}$ and so $1 - q = \frac{2}{3n-3}$. We have

$$X = \sum_{i=2}^{\infty} q^{i-2} i \tag{4}$$

and

$$qX = \sum_{i=2}^{\infty} q^{i-1} i = \sum_{i=3}^{\infty} q^{i-2}(i-1) \tag{5}$$

By subtracting Eq. 5 from Eq. 4, we have

$$\begin{aligned}
X - qX &= \sum_{i=2}^{\infty} q^{i-2} i - \sum_{i=3}^{\infty} q^{i-2}(i-1) \\
(1-q)X &= q^{2-2}(2) + \sum_{i=3}^{\infty} q^{i-2} i - q^{i-2}(i-1) \\
(1-q)X &= 2 + \sum_{i=3}^{\infty} q^{i-2} \\
(1-q)X &= 2 + \frac{q}{1-q} \\
\frac{2}{3n-3} X &= 2 + \frac{(\frac{3n-5}{3n-3})}{\frac{2}{3n-3}} \\
X &= \frac{3(3n^2 - 4n + 1)}{4}
\end{aligned}$$

Substitute X into $E_{max}(T_j)$, we have

$$\begin{aligned}
E_{max}(T_j) &= \frac{1}{n} + \frac{2}{3n} \cdot \frac{3(3n^2 - 4n + 1)}{4} \\
&= \frac{3n^2 - 4n + 3}{2n}
\end{aligned}$$

□

6 Simulation

In this section, we present our simulation study to evaluate our search strategies: 'circuit' and 'rebound'. The purposes of the simulation are: (i) to verify our theoretical results; and (ii) to evaluate the performance of different strategies in practice. A 'random' strategy [8] is implemented as the baseline method for comparisons. In the random strategy, each robot randomly goes to a neighboring cell and detects the target. All simulations are performed on a computer with i5 3.4 GHz CPU and 8 GB memory. The experimental platform is Matlab R2018b. All results are averaged over 10000 independently simulation executions.

6.1 Simulation Setup

We consider a 2D search area with size 40×40. The search area is divided into cells, each with size 1×1, i.e., there are 160 cells in total. There are three types of the targets as discussed in Sect. 5: (i) static target; (ii) mobile target; and (iii) superfast target. For type (ii) mobile targets, each target is assumed to move randomly. At each timeslot, it goes to a random direction at a random speed bounded by v_{max}, a preset maximum speed. In our simulation, we tested v_{max} in $\{0.25, 0.5, 0.75, 1\}$.

We tested the strategies in two scenarios: (i) single target; and (ii) multiple targets. All strategies were evaluated using the following performance metrics:

1. Achievement ratio: the ratio of simulation runs that find all targets within robots' lifetime.
2. Average latency: the average searching latency to find all targets upon a successful search.

6.2 Simulation Results on Single Target Searching

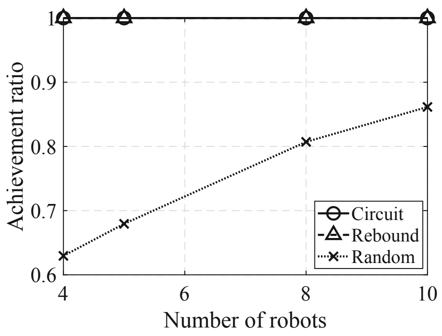
We performed the simulation for searching a single target varying the number of robots from 4 to 10. Figures 6 and 7 show the achievement ratios and average latency of different strategies in our simulations.

We make the following observations from the simulation results:

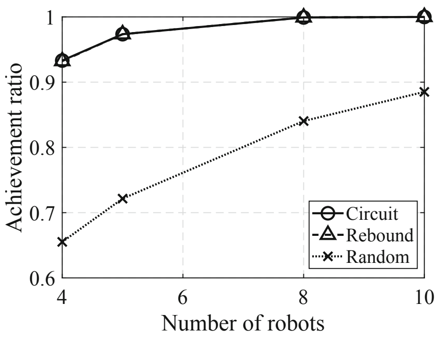
1. The circuit strategy has the same performance as rebound strategy for static and superfast targets while the circuit strategy is slightly better than the rebound strategy for mobile targets in all speeds. The circuit strategy is the best among the three in all scenarios.
2. The random strategy is the worst in all scenarios. The gap is significantly large except for superfast target (in which any strategy has the same chance to find the target for superfast target, as discussed in Theorem 2).
3. As the number of robots increases, the achievement ratio of the three strategies increases and the average latency decreases.
4. When the number of robots is 10, the achievement ratio of the circuit strategy and the rebound strategy is very close to 1.

6.3 Simulation Results on Multiple Targets Searching

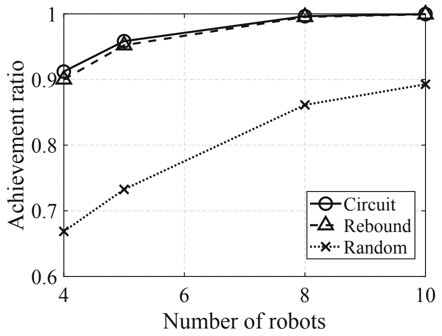
In this scenario, the number of robots is fixed to be 10. We performed the simulation for searching for 2 to 6 targets. The targets move independently.



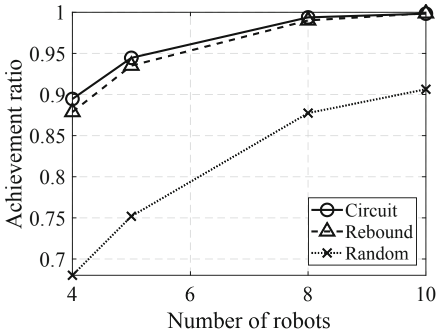
(a) Static target



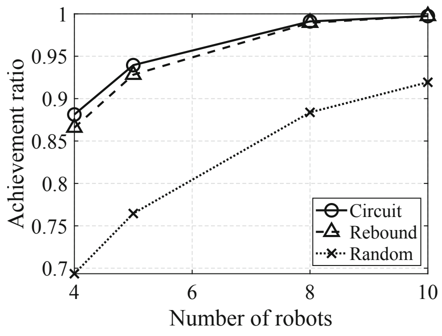
(b) Mobile target with $v_{max} = 0.25$



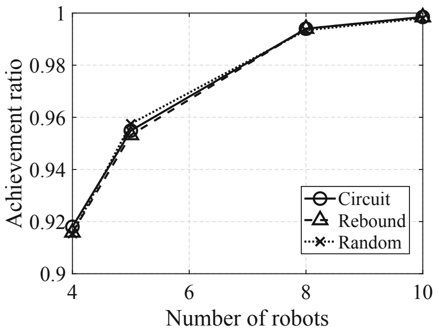
(c) Mobile target with $v_{max} = 0.5$



(d) Mobile target with $v_{max} = 0.75$



(e) Mobile target with $v_{max} = 1$



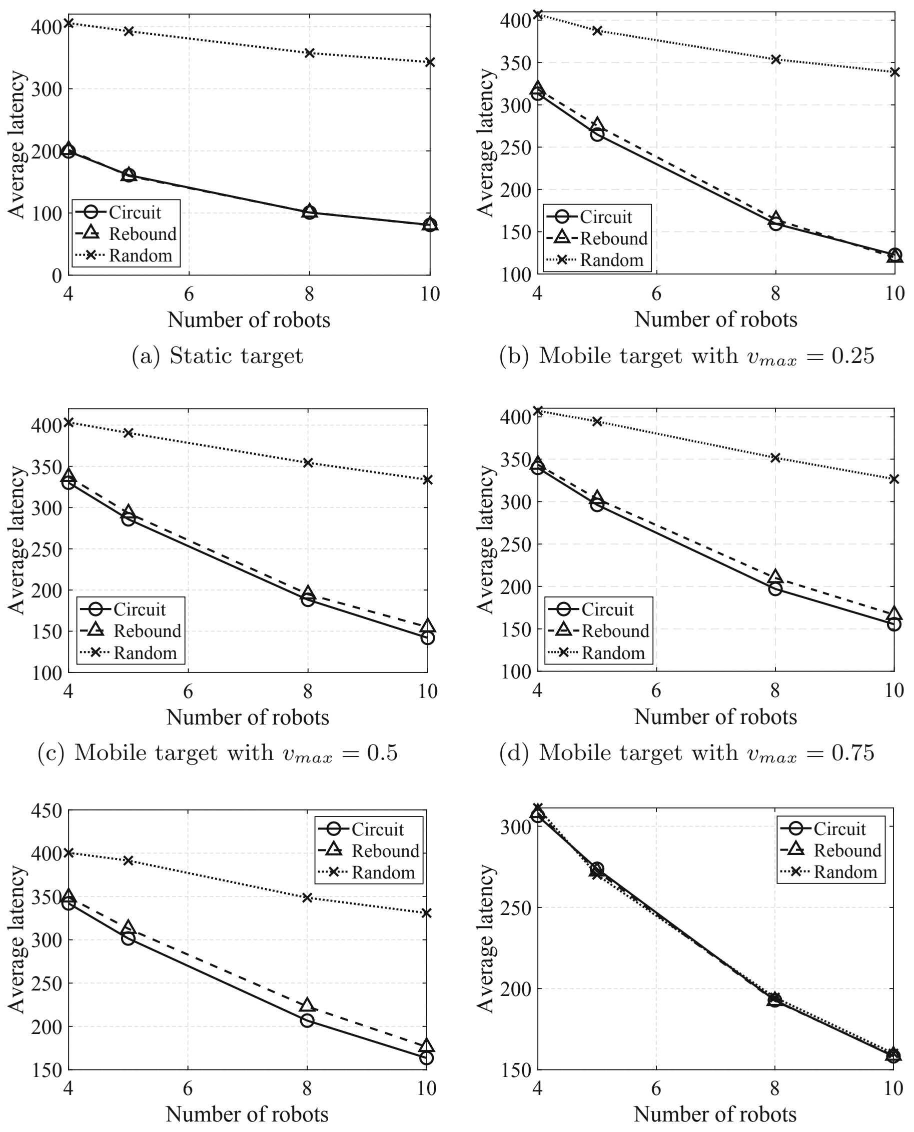
(f) Superfast target

Fig. 6. Achievement ratio of different strategies varying number of robots.

Figures 8 and 9 show the achievement ratios and average latency of different strategies in our simulations.

We make the following observations from the simulation results:

1. Similar to single target searching, the circuit strategy is the best among the three strategies. The circuit strategy and the rebound strategy have almost



(a) Static target (b) Mobile target with $v_{max} = 0.25$

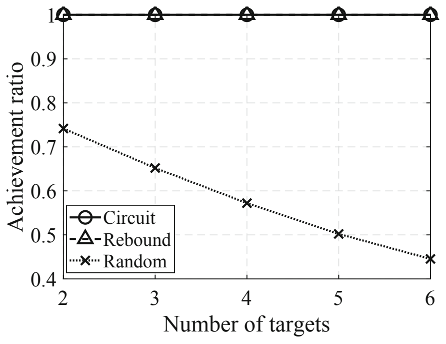
(c) Mobile target with $v_{max} = 0.5$ (d) Mobile target with $v_{max} = 0.75$

(e) Mobile target with $v_{max} = 1$ (f) Superfast target

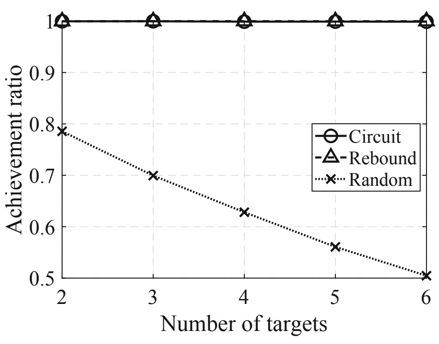
Fig. 7. Average latency of different strategies varying number of robots.

the same achievement ratio. Yet, the average latency of the circuit strategy is smaller than the rebound strategy for mobile targets (except for the case $v_{max} = 0.25$).

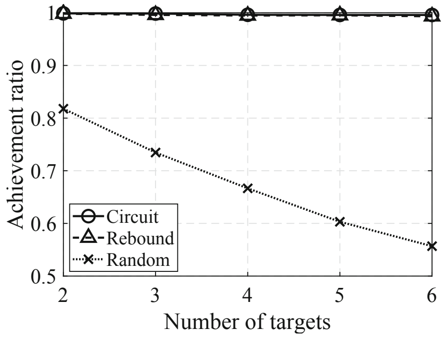
2. The average latency is not affected by number of targets. This is expected as the targets are independent.



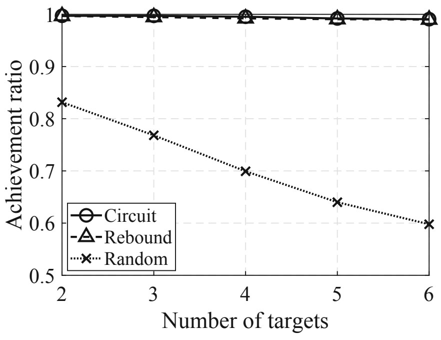
(a) Static targets



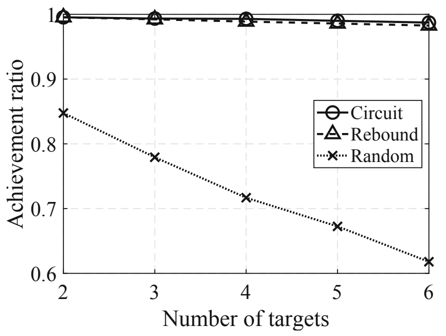
(b) Mobile targets with $v_{max} = 0.25$



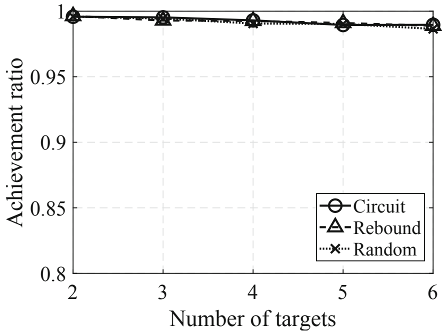
(c) Mobile targets with $v_{max} = 0.5$



(d) Mobile targets with $v_{max} = 0.75$



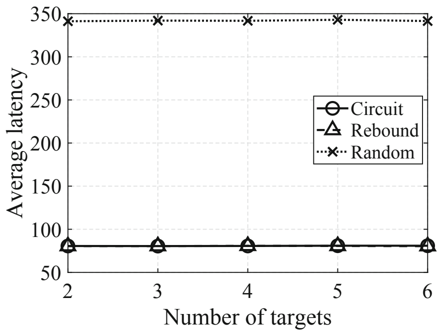
(e) Mobile targets with $v_{max} = 1$



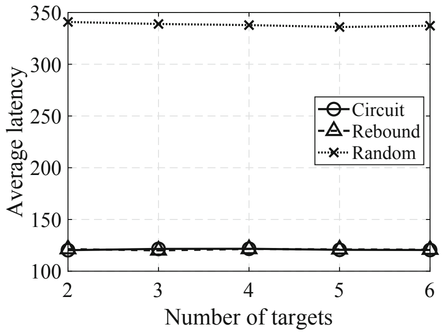
(f) Superfast targets

Fig. 8. Achievement ratio of different strategies varying number of targets.

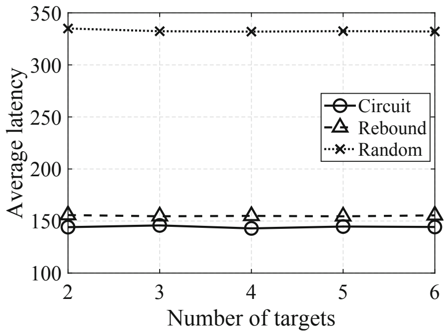
3. The random strategy is significantly worse than the circuit strategy and the rebound strategy in terms of both achievement ratio and average latency, except for case of superfast targets.



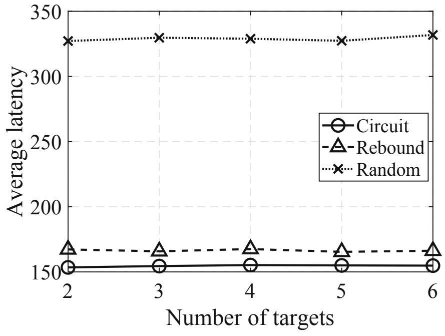
(a) Static targets



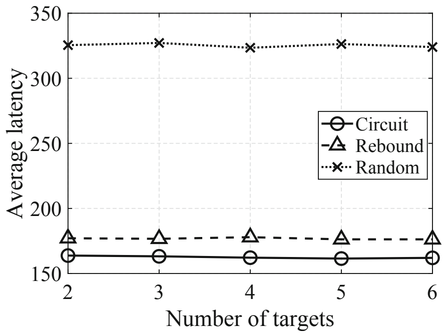
(b) Mobile targets with $v_{max} = 0.25$



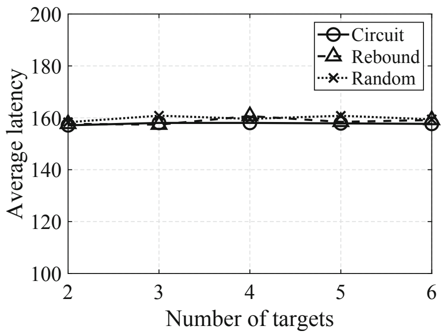
(c) Mobile targets with $v_{max} = 0.5$



(d) Mobile targets with $v_{max} = 0.75$



(e) Mobile targets with $v_{max} = 1$



(f) Superfast targets

Fig. 9. Average latency of different strategies varying number of targets.

4. As the number of targets increases, the achievement ratio of the random strategy drops significantly. The achievement ratios of the circuit strategy and the rebound strategy remain steady.

6.4 Discussions

All simulation results agree with our theoretical analysis. The reuslts show that the circuit strategy is the best in terms of achievement ratio and average latency. The performance of the rebound strategy is close to the circuit strategy but never outperforms the circuit strategy. The circuit strategy is suggested in practice for searching. In contrast, the random strategy is the worst and its performance is significantly worse than the other two strategies. This highlight the importance of picking the right strategy for searching in a disaster rescue mission.

7 Conclusions and Future Work

In this paper, we study the problem of mobile target searching using robots. No information about the targets is known. We propose a divide-and-conquer approach to divide the search areas into strips and let each robot search in one strip. We study two searching strategies, namely circuit strategy and rebound strategy. We theoretically analyze the searching latency in the strategies under different scenarios and conclude that the circuit strategy is better. The results are verified with extensive simulations. In the future, we plan to extend the model to allow the robots to communicate with each other during the search. This allows a more intelligent search strategy at the cost of increased communication cost and reduced lifetime of robots.

Acknowledgements. This work is partially supported by the Faculty Development Scheme (Ref. No. UGC/FDS14/E03/17 and UGC/FDS14/E01/17), The Deep Learning Research & Application Centre, and The Big Data & Artificial Intelligence Group in The Hang Seng University of Hong Kong.

References

1. DARPA Announces "Gremlins" UAS Program (2015). http://www.unmannedsystemstechnology.com/2015/09/darpa-announces-gremlins-uas-program/
2. Department of Defense Announces Successful Micro-Drone Demonstration, January 2017. https://www.defense.gov/News/News-Releases/News-Release-View/Article/1044811/department-of-defense-announces-successful-micro-drone-demonstration/
3. Celikkanat, H., Sahin, E.: Steering self-organized robot flocks through externally guided individuals. Neural Comput. Appl. **19**(6), 849–865 (2010)
4. Couzin, I.D., Jens, K., Franks, N.R., Levin, S.A.: Effective leadership and decision-making in animal groups on the move. Nature **433**(7025), 513–6 (2005)
5. Cucker, F., Dong, J.G.: Avoiding collisions in flocks. IEEE Trans. Autom. Control **55**(5), 1238–1243 (2010)
6. Delight, M., Ramakrishnan, S., Zambrano, T., MacCready, T.: Developing robotic swarms for ocean surface mapping. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 5309–5315, May 2016. https://doi.org/10.1109/ICRA.2016.7487742

7. Dell'Ariccia, G., Dell'Omo, G., Wolfer, D.P., Lipp, H.P.: Flock flying improves pigeons' homing: GPS track analysis of individual flyers versus small groups. Anim. Behav. **76**(4), 1165–1172 (2008)
8. Dimidov, C., Oriolo, G., Trianni, V.: Random walks in swarm robotics: an experiment with kilobots. In: Dorigo, M., et al. (eds.) ANTS 2016. LNCS, vol. 9882, pp. 185–196. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44427-7_16
9. Fang, H., Wei, Y., Chen, J., Xin, B.: Flocking of second-order multiagent systems with connectivity preservation based on algebraic connectivity estimation. IEEE Trans. Cybern. **47**(4), 1067–1077 (2017). https://doi.org/10.1109/TCYB.2016.2537307
10. Ferrante, E., Turgut, A.E., Stranieri, A., Pinciroli, C., Birattari, M., Dorigo, M.: A self-adaptive communication strategy for flocking in stationary and non-stationary environments. Nat. Comput. **13**(2), 225–245 (2014)
11. Fredette, D., Őzguner, U.: Swarm-inspired modeling of a highway system with stability analysis. IEEE Trans. Intell. Transp. Syst. **18**(6), 1371–1379 (2017). https://doi.org/10.1109/TITS.2016.2619266
12. de Marina, H.G., Jayawardhana, B., Cao, M.: Distributed rotational and translational maneuvering of rigid formations and their applications. IEEE Trans. Robot. **32**(3), 684–697 (2016). https://doi.org/10.1109/TRO.2016.2559511
13. Han, T., Ge, S.S.: Styled-velocity flocking of autonomous vehicles: a systematic design. IEEE Trans. Autom. Control **60**(8), 2015–2030 (2015). https://doi.org/10.1109/TAC.2015.2400664
14. Liu, H., Chu, X., Leung, Y.W., Du, R.: Simple movement control algorithm for bi-connectivity in robotic sensor networks. IEEE J. Sel. Areas Commun. **28**(7), 994–1005 (2010)
15. Michael, R., Alejandro, C., Radhika, N.: Robotics. Programmable self-assembly in a thousand-robot swarm. Science **345**(6198), 795–9 (2014)
16. Olfati-Saber, R., Jalalkamali, P.: Coupled distributed estimation and control for mobile sensor networks. IEEE Trans. Autom. Control **57**(10), 2609–2614 (2012). https://doi.org/10.1109/TAC.2012.2190184
17. Qiang, W., Li, W., Cao, X., Meng, Y.: Distributed flocking with biconnected topology for multi-agent systems. In: International Conference on Human System Interactions (2016)
18. Rango, F.D., Palmieri, N., Yang, X., Marano, S.: Swarm robotics in wireless distributed protocol design for coordinating robots involved in cooperative tasks. Soft. Comput. **22**(13), 4251–4266 (2018)
19. Sabattini, L., Chopra, N., Secchi, C.: Decentralized connectivity maintenance for cooperative control of mobile robotic systems. Int. J. Robot. Res. **32**(12), 1411–1423 (2013)
20. Sakthivelmurugan, E., Senthilkumar, G., Prithiviraj, K., Devraj, K.T.: Foraging behavior analysis of swarm robotics system. In: MATEC Web of Conferences, vol. 144, p. 01013. EDP Sciences (2018)
21. Semnani, S.H., Basir, O.A.: Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems. IEEE Trans. Cybern. **45**(1), 129–137 (2015). https://doi.org/10.1109/TCYB.2014.2328659
22. Szwaykowska, K., Romero, L.M., Schwartz, I.B.: Collective motions of heterogeneous swarms. IEEE Trans. Autom. Sci. Eng. **12**(3), 810–818 (2015). https://doi.org/10.1109/TASE.2015.2403253

23. Vásárhelyi, G., et al.: Outdoor flocking and formation flight with autonomous aerial robots. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3866–3873, September 2014. https://doi.org/10.1109/IROS.2014.6943105
24. Virágh, C., et al.: Flocking algorithm for autonomous flying robots. Bioinspir. Biomimet. **9**(2), 025012 (2013)
25. Ward, A.J.W., Herbert-Read, J.E., Sumpter, D.J.T., Jens, K.: Fast and accurate decisions through collective vigilance in fish shoals. Proc. Natl. Acad. Sci. U.S.A. **108**(6), 2312–2315 (2011)
26. Zhang, H., Chen, Z., Fan, M.: Collaborative control of multivehicle systems in diverse motion patterns. IEEE Trans. Control Syst. Technol. **24**(4), 1488–1494 (2016). https://doi.org/10.1109/TCST.2015.2487864
27. Zhao, H., Wang, H., Wu, W., Wei, J.: Deployment algorithms for uav airborne networks toward on-demand coverage. IEEE J. Sel. Areas Commun. **36**(9), 2015–2031 (2018). https://doi.org/10.1109/JSAC.2018.2864376
28. Zhao, H., Liu, H., Leung, Y.W., Chu, X.: Self-adaptive collective motion of swarm robots. IEEE Trans. Autom. Sci. Eng. **15**(4), 1533–1545 (2018)