Scheduling an Autonomous Robot Searching for Hidden Targets

B. Kriheli[1], E. Levner[1] , T.C.E. Cheng[2] and C.T. Ng[2]

[1]Department of Computer Science
Holon Institute of Technology, Holon, Israel
{borisk; levner}@hit.ac.il

[2]Department of Logistics and Maritime Studies
The Hong Kong Polytechnic University, Hong Kong SAR, China
{lgtcheng; lgtctng}@polyu.edu.hk

Abstract—The problem of searching for hidden or missing objects (called targets) by autonomous intelligent robots in an unknown environment arises in many applications, e.g., searching for and rescuing lost people after disasters in high-rise buildings, searching for fire sources and hazardous materials, etc. Until the target is found, it may cause loss or damage whose extent depends on the location of the target and the search duration. The problem is to efficiently schedule the robot's moves so as todetect the target as soon as possible. The autonomous mobile robot has no operator on board, as it is guided and totally controlled by on-board sensors and computer programs. We construct a mathematical model for the search process in an uncertain environment and provide a new fast algorithm for scheduling the activities of the robot which is used before an emergency evacuation of people after a disaster.

Keywords – intelligent robot; search-and-rescue; emergency evacuation; scheduling algorithm.

INTRODUCTION

During the last decades, research interest in planning, scheduling, and control of emergency response operations, especially people evacuation from high-rise and super high-rise buildings, has increased dramatically. In this paper we consider modeling and algorithmic aspects of the first component of the triad "detection-rescue-evacuation". A modern technology developed in recent years for solving the detection problem is the autonomous intelligent mobile robot equipped with on-board smart sensors and corresponding computer programs ([1]-[4]). Published research on the design of smart ("intelligent") robots and wireless sensor networks (WSNs) in emergency situations has recently grown exponentially.

Suppose a target like a fire or gas leakage has occurred somewhere in a large building. Until it is found and removed, the target may cause loss and damage to the whole building, the extent of which depends on the search time. The problem is to efficiently detect and remove the target as soon as possible with the help of an autonomous mobile robot (AMR) equipped with smart sensors and advanced communications systems. The AMR has no operator on board, i.e., it is fully autonomous, and is guided and controlled by on-board sensors and corresponding computer programs. The AMR can also be used for other detection-and-rescue missions that are dangerous, complicated or impossible for manned staff.

---

* This paper is a revised and extended version of the talk presented by the authors at the *2017 IEEE International Conference on Smart Computing (SMARTCOMP-2017)* (29-31 May 2017, Hong Kong) pp. 1-6, IEEE, 2017.

Typical objectives of the search problem are to maximize the probability of detection or to minimize the cost (or time) of the search (see, e.g., [5]-[8]). The problem of defining the best search strategy is fundamentally hard due to its probabilistic nature and combinatorial structure, and the nonlinearity induced by the detection probability. In particular, looking twice into the same location by a searcher does not necessarily double the detection probability. The corresponding combinatorial problem with general time and resource constraints is *NP*-hard ([9]). There are many published works providing different models, algorithms, and applications of the discrete search problem (see, e.g., [5]-[8], and the numerous citations in these works). In this paper we consider a simple but realistic setting happening in practice, namely we seek to minimize the incurred loss or damage with logistical and operational constraints limiting the search time.

In an unknown environment, the AMR's search-and-detection activity is subject to two types of error, namely a "false-negative" detection error wherein a target object is overlooked in spite that the robot is in a close neighborhood of the target, and a "false-positive" detection error, also called "a false alarm", in which a "clean" place or area is wrongly classified by the AMR as a target. As the general resource-constrained search problem is *NP*-hard, we restrict our study to finding local-optimal strategies under the specific scenario described below. We provide an algorithm for the AMR adopting a greedy search strategy in which, in each step, the on-board computer computes a current *search effectiveness* value for each location in the area and sequentially searches for a location with the highest search effectiveness value.

We organize the remainder of the paper as follows: The next section we review the related work. In Section III we present the search scenario and the mathematical model. In Section IV we analyze the problem and introduce the solution algorithm. In Sections V and VI we present a numerical example and discuss the computational results, respectively. In Section VII we conclude the paper.

II. PREVIOUS WORK

The optimal search problem is a well-known problem in Operations Research, Computer Science, and Artificial Intelligence (see, e.g., [5]-[8]). Many classical optimization techniques have been applied for solving various versions of the problem, such as, the Lagrange multipliers, mathematical programming, dynamic programming, game theory, branch-and-bound, and the local search. Our paper complements this stream of research. Note that our work is more selective, being restricted to discrete search by an autonomous device for a stationary target under imperfect inspections. Specifically, we apply a greedy solution method whereby robot's search moves are performed sequentially, step by step for each consecutive location. The greedy scheduling algorithm makes the AMR proceed by always searching, in the next step, a location with the highest search effectiveness value.

Given the chance that the AMR's inspection of a location containing the target may yield no definite information on whether or not the target is indeed in the inspected location, and, therefore the robot may visit the location (infinitely) many times, this type of scheduling problem may have an infinite sequence of steps in its solution process. Thus, the problem is to find *an optimal strategy* for the AMR to determine the next best search location dynamically, depending on the random parameters occurring up to the current step. Putting these next best search locations together will result in the overall locally optimal sequence.

A stopping rule in earlier works dictates the searcher to s*top in the step when the target is found for the first time*. However, this rule is obviously invalid when the positive-false error occurs. Other criteria for terminating the search process are treated in [5]-[8]; however, the termination conditions only correctly capture the false-negative search operations, i.e., under the assumption that the false-positive detection probabilities are zero. More detailed analysis of the early works can be found in the excellent texts [6]-[7].

In [10], the authors presented a Bayesian construction for the problem of searching for multiple targets by imperfect autonomous devices such as smart robots, using the probabilities of detection of these targets. [5] and [10] are the first works known to us wherein the authors addressed both false-negative and false-positive detection errors under the discrete search scenario. A major contribution of this work is the evaluation of several proposed (myopic and biology-inspired) search strategies minimizing the search time. Our paper complements that study in the case where the interference of a human verification team is expensive, complicated, or impossible, which leads to the case of exploiting autonomous smart robots.

Complementary to the search model in [10] in which the objective is to maximize the probability of detecting the target or to minimize the search time, we study another practical situation where no human team interference is involved, and consider the objective function of the "min-loss" type. We investigate a different search scenario and a stopping rule, develop a new greedy index-based search strategy, and establish its properties. In contrast to [6] and [10], our model does not require human interference. Specifically, the decision to stop the search is made automatically by the on-board computer program based on the input data and search history. The main contribution of the present work, in comparison with our previous works [11, 12], is that we introduce two types of the confidence levels for two error types, which permits us to reduce an infinite-horizon scheduling problem to a finite-horizon problem.

Under the scenario studied in this paper, we derive a locally optimal strategy. Being attractive because of its simplicity and computational efficiency, such a strategy guarantees the finding of an "almost optimal" search sequence with a given confidence level of target identification. Several earlier search models, in particular, the search model with only false-negative outcomes ([8], [11]) and the search model for perfect inspections ([13]) are special cases of our model.

III. PROBLEM FORMULATION AND MODEL

A search area of interest (AOI) is a complex area in a large smart home that contains a set of N possible locations (or "cells") of a stationary target. Given the stationary nature of the target, it does not leave the search area and new targets do not appear. The inspections by the AMR are imperfect. This means that there is a given prior probability α_i of a false alarm (if a target is falsely found when location i is clean) and, in addition, there is a given prior probability β_i of overlooking the target when actually location i contains a target, i =1,…, N. The imperfect quality of the inspections implies that examination of each location can happen more than once. Hence, the number of search steps, in general, may be infinite.

The durations and costs of the inspections of all the locations are known in advance and the goal of the search is to determine a search strategy for the AMR to employ in order to locate the target at the minimum cost.

The search area contains N cells or modules: 1, 2, …, N. The input data are the following:

- p_i - prior probability that location i contains a target;
- α_i - prior probability of a "false alarm", or a false-positive outcome; this is a conditional probability that a sensor of the AMR inspecting location i declares that the target is discovered in location i whereas, in fact, it is *not* in this location;
- β_i - prior probability of overlooking or a false-negative outcome; this is a conditional probability that the sensor of the AMR declares that the target is not in location i whereas, in fact, it is exactly there;
- t_i – expected time to inspect location i;
- c_i - search cost rate per unit time when searching location i; this is the amount (in monetary or physical units) of loss or damage incurred during one time unit of the search;
- CL^+ – a required positive confidence level, which is a pre-specified measure of the confidence that the target is identified correctly;

- CL^- – a required negative confidence level, which is a pre-specified measure of the confidence that the target is indeed not in the current location. The confidence level values are provided by experts and should be closed to 1; their roles will be discussed in more detail below.

The search strategy is defined as an infinite-horizon sequence of locations, where, in each step n, location $s[n]$ is tested as to whether it contains the target:

$s = <s[0], s[1], \ldots, s[n], \ldots>$.

Here $s(0)$ is an initial sub-sequence specified in advance by the human decision maker.

The problem is to find an optimal sequence s^{opt} of the search steps for detecting the target at the minimum total cost. We show below that, under the given confidence level values, it is sufficient to search for a certain finite sequence of steps s:

$s = <s[0], s[1], s[2], \ldots, s[M]>$,

where the finite value M is found depending on the input probabilities and the given confidence levels.

In order to exactly define the stopping rule of the search process, we first explain the notion of the confidence level. Without loss of generality, we suppose that in each step of s, say n, the on-board sensors examine a current location, say i, and the on-board computer can count and memorize the search history of the number of times, denoted by h_i, the sensors have already observed that location i contains the target (note that due to the presence of false positive outcomes, we are not obliged to immediately stop searching in that case). Similarly, by g_i we denote the number of times during the search process, up to a current step, the sensors have reported that the current location i does not contain the target.

Under our search scenario, the human team is not involved in verifying the search quality of the imperfect device. Instead, we consider a search process provided by a *completely autonomous* smart robot. In order to enhance automatically the search quality level, the on-board computer analyzes the input data and the overall search history, and then computes the following conditional probabilities $a_{[i,h_i]}$ and $b_{[i,g_i]}$:

- $a_{[i,h_i]}$, the conditional probability that the examined location (cell) i indeed contains the target under the condition that, during previous sequential inspections, the imperfect AMR's sensors have already declared h_i times that the cell i contains the target;

- $b_{[i,g_i]}$, the conditional probability that the examined location i does not contain the target under the condition that, during the previous sequential inspections, the imperfect AMR's sensors have repeatedly declared g_i times that the cell i does not contain the target.

The conditional probabilities $a_{[i,h_i]}$ and $b_{[i,g_i]}$ should be sufficiently high. Formally, we require that $a_{[i,h]}$ should not be less than a pre-specified value called the *positive confidence level* CL_i^+ (e.g., $CL_i^+ = 0.95$ or 0.99). A similar requirement is imposed on the probability $b_{[i,g_i]}$ and the negative confidence level. Below we study how the $a_{[i,h]}$ and $b_{[i,g]}$ values change when parameters h_i and g_i grow.

The confidence levels CL_i^+ and CL_i^- can be either equal or different for different locations. Without loss of generality, we suppose that the on-board computer of the AMR, in each step of s, say n, may count and memorize the search history of the number of times the sensors have already claimed, before the considered step, that location i contains (or does not contain) the target. These numbers accumulated up to step n (including that step) and associated with each location searched in that step in s are called the *heights* of the location $s[n] = i$ in step n and denoted by h_i and g_i, respectively. For notational simplicity, in the heights the index n is omitted. Notice that due to the existence of false alarms, the search should not stop when in step n the sensor declares for the first time that it discovers the target. We need to continue the search until the probability $a_{[i,h]}$ reaches (or exceeds) the pre-specified value CL_i^+.

Now we discuss the role of confidence levels in more detail. For any step n of *strategy* s, the *positive confidence level* CL_i^+ is defined as the desired lower bound for the conditional probability $a_{[i,h_i]}$ that location $s[n] = i$ indeed contains the target under the condition that the sensors have declared h_i times that the target is located in location i (i.e., this happens h_i -1 times in the previous steps 1,…, n-1 plus once more in the current step n). Similarly, the *negative confidence level* CL_i^- in step n is defined as the desired lower bound for the conditional probability $b_{[i,g_i]}$ that the sensors did not detect the target in location $s[n] = i$ under the condition that such a situation has happened totally g_i times in steps 1,2,…, n. The on-board computer can compute the number of times during their sequential inspections the AMR's sensors should declare that cell i *does* (*does not*) *contain* the target, so as to guarantee that $a_{[i,h_i]} \geq CL^+$ (and, respectively, $b_{[i,g_i]} \geq CL^-$).

The minimum positive integers h_i and g_i such that the probabilities $a_{[i,h_i]}$ and $b_{[i,g_i]}$ exceed or equal the corresponding confidence levels CL^+ and, CL^-, respectively, are called the *critical heights*, and denoted by H_i and G_i, respectively. They will be precisely calculated below. In what follows, for brevity's sake, the inspection of location i in which the sensor declared that it discovered the target for the h_i-th time (where $h_i = 1,\ldots, H_i$) is called the *h_i-th positive test.*

When the h_i value in step n of s is known, there is a direct way to compute the corresponding conditional probability $a_{[i,h_i]}$, for any h_i, and $b_{[i,g_i]}$, for any g_i as the following claim states.

Claim 1. The conditional probability $a_{[i,h_i]}$ and $b_{[i,g_i]}$ relate to their heights h_i and g_i, respectively, as follows:

$$a_{[i,h_i]} = \frac{p_i \cdot (1-\beta_i)^{h_i}}{p_i \cdot (1-\beta_i)^{h_i} + (1-p_i)\alpha_i^{h_i}}, \tag{1}$$

$$b_{[i,g_i]} = \frac{(1-p_i)\cdot(1-\alpha_i)^{g_i}}{(1-p_i)(1-\alpha_i)^{g_i} + p_i\beta_i^{g_i}}. \tag{2}$$

The proof is given in Appendix 1.

The search scenario is characterized by the following conditions:

(i) the cells are inspected sequentially and independently of one another;

(ii) for any target location, the outcomes of inspections are independent;

(iii) given a required value of the positive confidence level CL_i^+, the search terminates when, in some step n for some location $s[n] = i$ (and the corresponding height h_i), the probability $a_{[i,h_i]}$ achieves its requested confidence level, i.e.,

$$a_{[i,h_i]} \geq CL_i^+; \tag{3}$$

(iv) as soon as a pre-specified value of the negative confidence level CL_i^- for probability $b_{[i,g]}$ is reached, the corresponding cell i is considered "clean" and can be safely removed from further search, i.e.,

$$b_{[i,g_i]} \geq CL_i^-. \tag{4}$$

Claim 2. In each location i, the probabilities $a_{[i,h_i]}$ and $b_{[i,g_i]}$ grow monotonically with n, for any given values of α_i and β_i with $\alpha_i + \beta_i \leq 1$ (i.e., if the error probabilities α_i and β_i are "not too big").

The proof follows immediately from the following two facts: (a) the heights h_i and g_i of any location i increase with n in any search strategy s, and (b) the functions

$$a_{[i,h_i]}=\frac{p_i\cdot(1-\beta_i)^{h_i}}{p_i\cdot(1-\beta_i)^{h_i}+(1-p_i)\alpha_i^{h_i}}=\frac{1}{1+[(1-p_i)/p_i][\alpha_i/(1-\beta_i)]^{h_i}} \tag{5}$$

and

$$b_{[i,g_i]}=\frac{(1-p_i)\cdot(1-\alpha_i)^{g_i}}{(1-p_i)(1-\alpha_i)^{g_i}+p_i\beta_i^{g_i}}=\frac{1}{1+[(p_i)/(1-p_i)][\beta_i/(1-\alpha_i)]^{g_i}}$$

are increasing in h_i and g_i, respectively, whenever $\alpha_i/(1-\beta_i)\leq 1$.

By virtue of Claim 2, since the CL_i^+ and CL_i^- values are given in advance, the H_i and G_i values can be computed from (1)-(4) for all the locations before the AMR starts its search.

The logic of the considered scheduling scenario is the following. First, the positive and negative confidence levels, i.e., CL_i^+ and CL_i^-, respectively, for each location i are determined in advance, before the search starts, as the desired bounds for the corresponding probability to discover the target and the probability to discover that the location i does not contain the target, respectively. Then, knowing the CL_i^+ and CL_i^- values, and using (1)-(4), the robot's on-board computer finds the corresponding *critical heights* H_i and G_i (i.e., the minimum integer h_i and g_i satisfying, respectively, (3) and (4)), for any cell searched by the robot. After that, the greedy Algorithm A defined in the next section selects the locations to be searched one by one. Under our scenario, instead of having human involvement, the AMR stops its search automatically when, for some cell i, the outcome of the current inspection is as follows: "for the H_i -th time cell i is discovered to contain the target". Note that in this case the probability of the correct decision is guaranteed to be sufficiently high, without additional human-made testing. On the other hand, as soon as the height g_i achieves its critical value G_i, we accept that this location is clean and can be removed from further search.

Now we are able to precisely formulate the stopping rule.

The search process stops in the step when either (i) the number of positive tests provided by the sensors for some location, say i, reaches, for the first time, its critical height value H_i or (ii) all the cells are deleted from the search due to the negative tests. Therefore, our model does not require human operator's interference for defining the termination moment. It is worth noticing that, due to the presence of the confidence levels, as soon as the robot visits any cell, say i, at most H_i + G_i -1 times, either the algorithm terminates the search or such a cell i is removed. Thus, under our premises, the search process is finite and the total number of steps does not exceed $\sum_i (H_i+G_i-1)$.

IV. PROBLEM ANALYSIS

In the following analysis we need the following notation:

- $T_{s[n]} = T(s[n], s)$ - (accumulated) time spent by the AMR when moving from $s[1]$ to location s[n] in the n-th step of strategy s; $T_{s[n]} = \sum_{m=1}^{n} t_{s[m]}$,
- $F(s)$ - the expected total loss incurred by the target before the latter is found and neutralized.

We compute the expected total loss $F(s)$ as the mathematical expectation in the space of the following conditional probabilities defined in each step of the search:

- $P_{s[1]} = \mathrm{P}(h_{s[1]} = H_{s[1]})$ - probability to discover the target in the first step of the search;
- $P_{s[2]} = \mathrm{P}(h_{s[2]} = H_{s[2]} \mid h_{s[1]} < H_{s[2]})$ - probability to discover the target in the second step of the search under the condition that the robot did not discover the target in the cell visited in the first step;

  … ;

- $P_{s[n]} = \mathrm{P}(h_{s[n]} = H_{s[n]} \mid h_{s[k]} < H_{s[k]}$ for $k = 1,\ldots, n-1)$ - conditional probability that in the sequence $<s[1], s[2], \ldots, s[n-1]. s[n]>$ in location $s[n]$ the robot has discovered the target for the $H_{s[n]}$-th time (so the on-board computer commands the robot to stop at this step) under the condition that the robot did not discover the target in any cell in its previous steps in sequence $<s[1], s[2], \ldots, s[n-1]>$ (i.e., under the condition that the robot does not stop in any cell prior to step n).

(The values $H_{s[n]}$ depending on the parameters p_i, α_i, and β_i, and guaranteeing the required confidence level CL^+ are computed in the previous section, and the $P_{s[\mathrm{n}]}$ is computed below.)

In accordance with conditions (i) and (ii) of the considered search scenario given in the previous section, the expected (linear) total loss $F(s)$ is defined as follows:

$$F(s) = \sum_{n=1}^{\infty} P_{s[n]} c_{s[n]} T_{s[n]}.$$

In the above notation, we wish to solve the finite-horizon search problem under uncertain inspection data, i.e., to find a finite sequence s^{opt} of the search steps minimizing the expected total loss $F(s)$ incurred by the target before it is found: $s^{opt} = <s[0], s[1], s[2], \ldots, s[M]>$. As noticed above, under the given confidence levels, the total number of steps, denoted by M, does not exceed $\sum_i (H_i + G_i - 1)$.

This scheme is an extension, for the case of two types of possible errors, of the classic infinite-horizon (single-error-type) search analysis by De Groot [14].

For any given sequence s and the location $s[n]$ inspected by the sensors in step n, let $A_{s[n]}$ be the total number of inspections of the location $s[n]$ (not necessarily successively) up to the nth step of strategy s.

Claim 3. The probability $P_{s[n]}$ is computed as follows:

$$P_{s[n]} = \binom{A_{s[n]}-1}{H_{s[n]}-1}(1-f_{s[n]})^{A_{s[n]}-H_{s[n]}}\left(f_{s[n]}\right)^{H_{s[n]}},$$ for $A_{s[n]} \geq H_{s[n]}$ and $h_{s[i]} < H_{s[i]}$ for all i = 1, …, n-1; and $P_{s[n]} = 0$, otherwise.

This claim immediately follows from the above definitions of $H_{s[n]}$ and $A_{s[n]}$. Indeed, the robotic search under investigation can be re-stated as a sequence of independent Bernoulli trials such that a "success" corresponds to the event that the robot discovers the target during a single inspection. It is known that the probability of $H_{s[n]}$ successes in a location $s[n]$ within the total number $A_{s[n]}$ of inspections of the location $s[n]$ occurring during n trials, i.e., n steps of s, has a negative binomial distribution (***for the corresponding definition*** and details, we refer the reader to [15] –[18].

Define the ratio $Q_{s[n]}$ as follows:

$$Q_{s[n]} = (c_{s[n]}/t_{s[n]})\binom{A_{s[n]}-1}{H_{s[n]}-1}\left(1-f_{s[n]}\right)^{A_{s[n]}-H_{s[n]}} \cdot f_{s[n]}^{H_{s[n]}}. \quad (5)$$

For any location i, the above ratio can be re-written as $Q_i = (c_i/t_i)P_i$, representing a measure of "attractiveness" of each location from the searcher's point of view, which is proportional to the probability of successful detection and the search value, and inversely proportional to the search time.

Algorithm A. The greedy fast algorithm A is constructed as follows: The robot searches the locations sequentially in the search sequence s* in such a way that the ratio $Q_{s^*[n]}$ has the maximum possible value in each step for all $n \geq 1$.

The key result of this work is formulated as follows:

Theorem. The strategy s is locally optimal if, in each step n, the location with the maximum ratio

$$Q_{s[n]} = (c_{s[n]} / t_{s[n]}) \binom{A_{s[n]} - 1}{H_{s[n]} - 1} \left(1 - f_{s[n]}\right)^{A_{s[n]} - H_{s[n]}} \cdot f_{s[n]}^{H_{s[n]}}$$

$n \geq 1$

is selected among all the possible locations. That is, the location $s[n] = i$ is selected such that

$$Q_{s[n]=i} = \max_{1 \leq j \leq N} Q_{s[n]=j}, \quad n \geq 1.$$

The proof is given in Appendix 2.

The search procedure in the theorem is greedy, or index-based, so far as in each step the on-board search program selects for inspection the next maximum-ratio location. Notice that under the considered scenario the on-board computer uses only partial information about the history of all the search outcomes. A numerical example illustrating the work of the algorithm is given in the next section.

V. NUMERICAL EXAMPLE

Consider an example with N = 5 locations. The input data are given in Table 1. The confidence levels equal 0.95.

Table 1. Input data

Location	1	2	3	4	5
$p_i = P(C_i)$	0.2	0.45	0.75	0.6	0.25
$\beta_i = P(\overline{B}_i / C_i)$	0.4	0.07	0.05	0.03	0.3
$\alpha_i = P(B_i / \overline{C}_i)$	0.04	0.06	0.12	0.2	0.1
t_i (min)	5	8	10	7	8
c_i (\$)	3	5	10	3	6

The probabilities f_i and critical heights H_i and G_i are computed using (1)-(4) and presented in Table 2.

Table 2. Auxiliary parameters

f_i	0.152	0.4515	0.7425	0.662	0.25
H_i	2	2	3	2	3
G_i	2	2	2	2	2

The initial sub-sequence $U_{[s,0]}$ is defined by the expert, e.g., according to the following rule:

$$U_{[s,0]} = \left\{ \underbrace{1,1,,,1}_{H_1-1}, \underbrace{2,2,...2}_{H_2-1}, ..., \underbrace{N,N,...N}_{H_N-1} \right\}.$$

Hence, in this example, $U_{[s,0]}$ defines the first seven steps: $U_{[s,0]} = \{1, 2, 3, 3, 4, 5, 5\}$.

In the next step, $n = 8$, using (5), we obtain the following *ratio* candidates Q: $Q_1 = 0.43146$, $Q_2 = 0.18804$, $Q_3 = 0.01707$, $Q_4 = 0.04896$, and $Q_5 = 0.31640$. Therefore, the location to be selected after $U_{[s,0]}$ is location 1: $s[8] = 1$. Continuing the process in the same way (and omitting the intermediate calculations), we obtain: $s[9] = 5$ and then $s[10] = 5$.

A non-trivial situation is encountered at this stage: Up to this step, the greedy algorithm dictates to the robot the following sequence of locations to be searched: s = $\{1,2,3,3,4,5,5,1,5,5\}$. After that the robot inspects location 5 (we see that this is, in fact, the fourth sequential inspection of this location in the obtained sequence), one of two possible outcomes can occur in the uncertain environment: either (a) the sensors claim for the third time that location 5 contains the target; in this case the search can be stopped, with the confidence level being successfully achieved; or (b) during the searches, the sensors twice discovered that location 5 does not contain the target; it follows that within the given (negative) confidence level value $CL_5^- = 0.95$, location 5 can be removed from further search, and the search process is continued with the remaining locations.

Continuing computation in the same way (with the remaining locations), comparing the Q values, we find that $s[11]$ = 2 and then $s[12]$ = 2. Up to this step, location 2 turns out to be inspected three times. Two options are possible: either the sensors declared twice that the target is in location 2, in which case we accept that the robot discovered the target with the given level of confidence and the search can be stopped in this step 12; or the sensors declared twice that location 2 does not contain the target, in which case location 2 can be removed from further search, and the search process is continued with the remaining locations.

In the next step, comparing the Q values, we find that $s[13]$ = 1. Up to this step, location 1 is inspected three times. Two options are possible: either the sensors declared twice that the target is in location 1, in which case we accept that the robot discovered the target with the given level of confidence and the search can be stopped in this step 13; or the sensors declared twice that location 1 does not contain the target, in which case the location can be removed from further search, and the search process is continued with the remaining locations.

In the next step, comparing the Q values, we find that $s[14]$ = 4 and then $s[15]$ = 4. Up to this step, location 4 is inspected three times. Two options are possible: either the sensors declared twice that the target is in location 4, in which case we accept that the robot discovered the target with the given level of confidence and the search can be stopped in this step 15; or the sensors declared twice that location 4 does not contain the target, in which case the location can be removed from further search, and the search process is continued with the remaining locations. In the next step, comparing the Q values for the remaining locations, we find that $s[16]$ = 3 and after that $s[17]$ = 3. We observe that, up to this step, location 3 is inspected 4 times. Two options are possible: either the sensors declared twice that the target is in location 3, in which case we accept that the robot has discovered the target in location 3 with the given level of confidence, and the search can be stopped; or the sensors declared twice that location 4 does not contain the target, in which case the location can be removed from further search, and the search process may be continued for the remaining locations. However, in this example, no locations are left, and the search stops. So we arrive at the outcome that, for the given set of input data, including the confidence levels, the robot cannot find the target. In this case, if needed, the human decision maker can change the input data and repeat the search.

For the reader's convenience, we present Table 1 that summarizes the above computation. The locations inspected during the first seven steps of the optimal sequence are provided by $U_{[s,0]} = \{1, 2, 3, 3, 4, 5, 5\}$. The next ten locations are presented in Table 1.

Table 1. The locations inspected in steps 1 to 17.

Step	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Location	1	2	3	3	4	5	5	1	5	5	2	2	1	4	4	3	3

The suggested search algorithm quickly terminates. The following locally optimal search sequence is obtained, in at most 17 steps: $s^{opt}=\{1,2,3,3,4,5,5,1,5,5,2,2,1,4,4.3,3\}$ and $F(s^{opt})$ = \$1,275.8.

We compare the obtained solution with another solution, obtained by using a manual greedy search strategy of selection according to the maximum probability of detecting the target in each step. In this numerical example, we obtain s^0 = <551133344225511> with $F(s^0)$=1,559.4 (K\$), i.e., the proposed algorithm essentially outperforms the manual algorithm.

VI. COMPUTATIONAL RESULTS

We implemented the developed greedy algorithm using MATLAB v. 7.1 and executed it on the following hardware platform: Dell PowerEdge R410 server, CPU - 2x Intel Xeon X5660 @ 2.8GHz (12 cores total), and 64 GB RAM. We randomly generated the probabilities in the interval (0.2, 0.9).

We carried out an extensive computational experiment over a total of 2,000 different randomly generated problem instances. The calculation time was limited to 0.01 CPU-hour per problem instance. Table 2 shows the characteristics of input data.

Table 2. The input data characteristics.

Data Type	Range
Number of locations	Unif (10,40)
Search time (in min.)	Unif (1,20)

Location probabilities	Unif (0.01, 0.90)
Probabilities of errors (type I and II)	Unif (0.01, 0.04)
Probabilities of overlooking	Unif (0.02, 0.04)
Search costs	Unif (10,30)

For each instance, we compared two different solutions produced by the solver: a locally optimal solution yielded by the suggested algorithm and a best solution generated by the full search over 1,000 randomly generated feasible solutions for each instance. The following fact similar to the well-known "80/20 Pareto rule" (see [18]) has been empirically observed, according to which for about 80% of the instances solved by our locally optimal algorithm the average cost is within 20% of the best value produced by the random search.

Figure 1 gives the results for different average cost values (from $10 to $500).

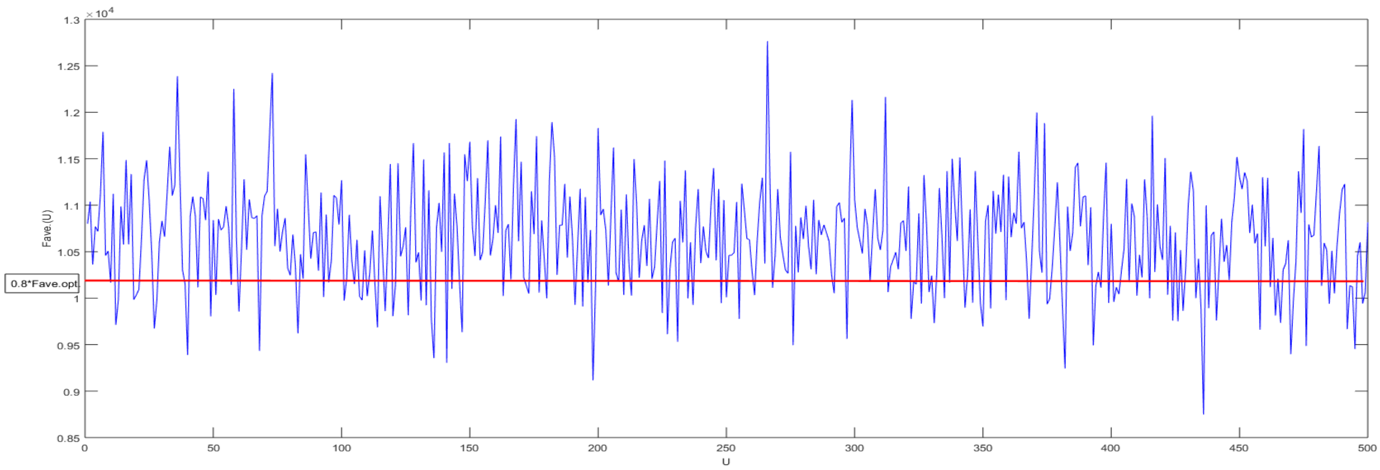

Figure 1. The average number of strategies yielding a better solution than the greedy solution.

VII. CONCLUSIONS

The suggested greedy algorithm is simple, user-friendly, and computationally efficient. Not only does it provide a good enough cost of the search, it is extremely fast and practicable. Although deploying search-and-rescue robots in the urban and building environment is still in its infancy, the first use of smart robots for search and rescue in emergency evacuation was successful ([1]-[4]). In contrast to many other stochastic search scenarios under uncertain inspection in which the search sequence is infinite, the suggested scenario, for any given confidence levels, has a finite number of steps.

In our future research we will extend the suggested approach to cover a wider range of practical search scenarios (e.g., with multiple mobile robots and multiple targets). It could be combined with more general solution methods such as dynamic programming, branch-and-bound, and nature-inspired algorithms. Another important future research topic is to embed into and integrate the proposed analysis with a general model of the triad "search-rescue-evacuation".

ACKNOWLEDGMENT

The authors wish to thank the Editor and anonymous reviewers for their very useful comments and suggestions. This research was supported in part by the Research Grants Council of Hong Kong under grant no. PolyU 152148/15E.

REFERENCES

[1] J. Casper and R.R. Murphy, 2003. "Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center". *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3), pp.367-385, 2003.

[2] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, G. Pereira, and J. Spletzer, "Distributed search and rescue with robot and sensor teams". In *Field and Service Robotics*, Springer, pp. 529-538, 2006.

[3] R.R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A.M. Erkmen, "Search and rescue robotics". In *Springer Handbook of Robotics* 2008 (pp. 1151-1173), Springer Berlin Heidelberg.

[4] H.-H. Hsu, J.-K. Chang, W.-J. Peng, T.K. Shih, T.W. Pai, and K.L. Man, "Indoor localization and navigation using smartphone sensory data". *Annals of Operations Research*, 265(2), pp 187–204, 2018.

[5] R.F. Dell, J.N. Eagle, G.H.A. Martins, and A.G. Santos, "Using multiple searchers in constrained-path, moving-target search problems". *Naval Research Logistics* 43, pp. 463-480, 1996.

[6] M. Kress, K.Y. Lin, and R. Szechtman, "Optimal discrete search with imperfect specificity", *Mathematical Methods of Operations Research* 68, pp. 539-549, 2008.

[7] L.D. Stone, *Theory of Optimal Search*. Academic Press, New York, 1989.

[8] S.J. Benkoski, M.G. Monticino, and J.R. Weisinger,"A survey of the search theory literature". *Naval Research Logistics* 38, pp. 469-494, 1991.

[9] [K.E. Trummel and J.R. Weisinger, "The complexity of the optimal searcher path problem". *Operations Research* 34. pp. 324-327, 1986.

[10] T.H. Chung and J.W. Burdick, "Analysis of search decision making using probabilistic search". *IEEE Transactions on Robotics* 28, pp. 132-144, 2012.

[11] E. Levner, "Infinite-horizon scheduling algorithms for optimal search for hidden objects". *International Transactions on Operational Research* 1, pp. 241-250, 1994.

[12] B. Kriheli, E. Levner, and A. Spivak, "Optimal search for hidden targets by unmanned aerial vehicles under imperfect inspections". *American Journal of Operations Research* 26, 6(2), pp. 53-66, 2016.

[13] G. Rabinowitz, and H. Emmons, "Optimal and heuristic inspection schedules for multistage production systems". *IIE Transactions*, 29 (12), 1997.

[14] M.H. De Groot, *Optimal Statistical Decisions*, Chapter14, 1970.

[15] M.H. De Groot,. *Probability and Statistics,* Addison-Wesley. 1978 pp. 258–259. *ISBN 0-201-11366-X*.

[16] R.C. Tripathi, "Negative binomial distribution". *Encyclopedia of Statistical Sciences*, 2006.

[17] A. Ninh and M. Pham, "Logconcavity, twice-logconcavity and Turán-type inequalities", *Annals of Operations Research*, 2018, pp.1-13, https://doi.org/10.1007/s10479-018-2923-y.

18] A. Rushton, J. Oxley and P. Croucher, *The Handbook of Logistics and Distribution Management*, Kogan Page, London, 2000, pp.107-108.

Appendix 1. Proof of Claim 1.

Let us introduce the following additional notation:

Event B_i = {after a single inspection, the AMR's sensors declare that location i contains the target}.

Event C_i ={location i indeed contains the target}.

In terms of these events, the location probabilities p_i and the probabilities of the two types of error are expressed as follows:

$$p_i = P(C_i);\ \ \alpha_i = P(B_i / \overline{C}_i);\ \ \beta_i = P(\overline{B}_i / C_i).$$

The probability f_i that the sensor declares that location i is detected as containing the target is

$$f_i = P(B_i) = P(B_i / C_i)P(C_i) + P(B_i / \overline{C}_i)P(\overline{C}_i) = (1- \beta_i)\, p_i + \alpha_i(1- p_i).$$

The conditional probability of the event that location i contains the target under the condition that the sensor has declared that location i contains the target in a single inspection is

$$P(C_i / B_i) = \frac{P(C_i)P(B_i / C_i)}{P(C_i)P(B_i / C_i) + P(\overline{C}_i)P(B_i / \overline{C}_i)} = \frac{p_i \cdot (1-\beta_i)}{p_i \cdot (1-\beta_i) + (1-p_i)\alpha_i}.$$

We need to prove that the conditional probability $a_{[i,h_i]}$ of the event that location *i* indeed contains the target under the condition that the sensor has declared in exactly h_i steps of the sequence *s* that location *i* contains the target satisfies the following relation:

$$a_{[i,h_i]} = P\left(C_i / B_i^{(1)} \cap B_i^{(2)} \cap \ldots \cap B_i^{(h_i)}\right) = \frac{p_i \cdot (1-\beta_i)^{h_i}}{p_i \cdot (1-\beta_i)^{h_i} + (1-p_i)\alpha_i^{h_i}}.$$

Since the sequential inspections of locations made by the robot are independent, for any pair of indices *i*, $j \in \{1,2,\ldots,N\}$, we have the following equalities:

$$\begin{cases} P\left(B_i^{(k+1)} / C_i \cap B_j^{(k)}\right) = P\left(B_i^{(k+1)} / C_i\right) \\ P\left(B_i^{(k+1)} / \overline{C}_i \cap B_j^{(k)}\right) = P\left(B_i^{(k+1)} / \overline{C}_i\right) \end{cases}.$$

Therefore,

$$a_{[i,h_i]} = \frac{P(C_i)\cdot P\left(B_i^{(1)} \cap B_i^{(2)} \cap \ldots \cap B_i^{(h_i)} / C_i\right)}{P\left(B_i^{(1)} \cap B_i^{(2)} \cap \ldots \cap B_i^{(h_i)}\right)} = \frac{P(C_i)\cdot P^{h_i}\left(B_i^{(1)} / C_i\right)}{P(C_i)\cdot P^{h_i}\left(B_i^{(1)} / C_i\right) + P(\overline{C}_i)\cdot P^{h_i}\left(B_i^{(1)} / \overline{C}_i\right)} =$$

$$\frac{p_i \cdot (1-\beta_i)^{h_i}}{p_i \cdot (1-\beta_i)^{h_i} + (1-p_i)\alpha_i^{h_i}}.$$

The second part of the Claim 1 is proved along the same line.

Appendix 2. Proof of the Theorem

$$s_1 = U_{[s_1,0]}, s_1[1], s_1[2], \ldots, s_1[n], s_1[n+1], \ldots$$

$$s_2 = U_{[s_1,0]}, s_1[1], s_1[2], \ldots, s_1[n+1], s_1[n], \ldots$$

To prove the theorem, it suffices to prove the following relation:

$$F(s_1) \le F(s_2) \Leftrightarrow Q_{s_1[n]} \ge Q_{s_1[n+1]}.$$

We have:

$$F(s_1) = \sum_{1 \le m \le M} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} = \sum_{1 \le m \le M} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} \left(A_{s_1[m]} \right) =$$

$$= \sum_{m=1}^{n-1} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} + \sum_{n+2 \le m \le M} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} +$$

$$+ c_{s_1[n]} \cdot T_{s_1[n]} \cdot P_{s_1[n]} + c_{s_1[n+1]} \cdot T_{s_1[n+1]} \cdot P_{s_1[n+1]}$$

$$F(s_2) = \sum_{1 \le m \le M} c_{s_2[m]} \cdot T_{s_2[m]} \cdot P_{s_2[m]} = \sum_{1 \le m \le M} c_{s_2[m]} \cdot T_{s_2[m]} \cdot P_{s_2[m]} \left(A_{s_2[m]} \right) =$$

$$= \sum_{m=1}^{n-1} c_{s_2[m]} \cdot T_{s_2[m]} \cdot P_{s_2[m]} + \sum_{n+2 \le m \le M} c_{s_2[m]} \cdot T_{s_2[m]} \cdot P_{s_2[m]} +$$

$$+ c_{s_2[n]} \cdot T_{s_2[n]} \cdot P_{s_2[n]} + c_{s_2[n+1]} \cdot T_{s_2[n+1]} \cdot P_{s_2[n+1]}$$

$$F(s_2) = \sum_{m=1}^{n-1} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} + \sum_{n+2 \le m \le M} c_{s_1[m]} \cdot T_{s_1[m]} \cdot P_{s_1[m]} +$$

$$+ c_{s_1[n+1]} \cdot \left(T_{s_1[n+1]} - t_{s_1[n]} \right) \cdot P_{s_1[n+1]} + c_{s_1[n]} \cdot T_{s_1[n+1]} \cdot P_{s_1[n]}$$

$$F(s_1) - F(s_2) =$$

$$= c_{s_1[n]} \cdot T_{s_1[n]} \cdot P_{s_1[n]} + c_{s_1[n+1]} \cdot T_{s_1[n+1]} \cdot P_{s_1[n+1]} -$$

$$- c_{s_1[n+1]} \cdot \left(T_{s_1[n+1]} - t_{s_1[n]} \right) \cdot P_{s_1[n+1]} - c_{s_1[n]} \cdot T_{s_1[n+1]} \cdot P_{s_1[n]} =$$

$$= -c_{s_1[n]} \cdot P_{s_1[n]} \cdot \left(T_{s_1[n+1]} - T_{s_1[n]} \right) + c_{s_1[n+1]} \cdot t_{s_1[n]} \cdot P_{s_1[n+1]} =$$

$$= -c_{s_1[n]} \cdot P_{s_1[n]} \cdot t_{s_1[n+1]} + c_{s_1[n+1]} \cdot t_{s_1[n]} \cdot P_{s_1[n+1]}$$

We obtain that

$$F(s_1) - F(s_2) = -c_{s_1[n]} \cdot P_{s_1[n]} \cdot t_{s_1[n+1]} + c_{s_1[n+1]} \cdot t_{s_1[n]} \cdot P_{s_1[n+1]}$$

$$F(s_1) \le F(s_2) \Leftrightarrow F(s_1) - F(s_2) \le 0 \Leftrightarrow$$

$$-c_{s_1[n]} \cdot P_{s_1[n]} \cdot t_{s_1[n+1]} + c_{s_1[n+1]} \cdot t_{s_1[n]} \cdot P_{s_1[n+1]} \le 0 \Leftrightarrow$$

$$c_{s_1[n+1]} \cdot t_{s_1[n]} \cdot P_{s_1[n+1]} \le c_{s_1[n]} \cdot P_{s_1[n]} \cdot t_{s_1[n+1]} \Leftrightarrow$$

$$\frac{c_{s_1[n+1]} \cdot P_{s_1[n+1]}}{t_{s_1[n+1]}} \le \frac{c_{s_1[n]} \cdot P_{s_1[n]}}{t_{s_1[n]}} \Leftrightarrow$$

$$Q_{s_1[n+1]} \le Q_{s_1[n]} \Leftrightarrow$$

$$Q_{s_1[n]} \ge Q_{s_1[n+1]}$$

The theorem is proved.