Developing a Test-bed for Distributed Search by Mobile Sensors

Katharina Ley, Kevin Comer, Julian Counihan, David Czarnecki, Sarah Foster, Huma Husain, Rebecca McCrabb

Abstract-- The goal of this Systems and Information Engineering Capstone project is to develop a physical test-bed in order to analyze the efficiency of swarm-based distributed search algorithms. This test-bed demonstrates a distributed search system using actual moving search agents and target detection. The goal of this test-bed is to provide a tool that can be used in future research to better understand swarm-based distributed search by mobile sensors.

I. INTRODUCTION

Automated searches are used for a wide array of different tasks in a broad variety of fields. Search vehicles are used to locate landmines in dangerous mine fields, as well as to find mineral deposits, craters, or traces of water on far-away planets and asteroids. In all of these scenarios, precise methods of automated target location could be the difference between mission failure and success. Distributed search is especially well suited for these applications, since this type of search minimizes communications and data storage requirements. It is costly or even impossible, for example, to support constant communications between robots on other planets and a home base here on earth, thus making centralized schemes unviable. Additionally, extensive bilateral communication requirements could compromise the scalability of a search.

The main goal of this Capstone Project is to design a test-bed that may be used to assess the viability of specific algorithms used for automated searches. A search algorithm simply refers to a logical framework that allows search agents to cover a search grid. A search grid can be any area in which targets may be located, subdivided into a grid of potential target locations. Agents then move about the search grid probing each target location to determine whether a target is present [1].

II. PROBLEM DEFINITION

The mission of this Capstone project can be broken down into two main components. The first phase is comprised of the development of a physical model of a search grid and search robots. Specifications for this system include a reliable localization system for search agents, a method for accurately and remotely controlling agent movement, and a way to simulate the agent's probing for targets. The second component of the project is comprised of developing software that will allow the test-bed to simulate a distributed search. This software allows agents to make movement decisions that will optimize the search based upon a specific search algorithm.

III. BACKGROUND & RATIONALE

This Capstone Project builds upon the work by Garcia et al (2005) and Beling et al. (2005) and (2006). In particular, we focus on a distributed search algorithm based on the natural behavior of swarming ants. Distributed algorithms are particularly efficient because each search agent independently chooses where to search for a target next. In contrast, centralized searches are controlled by a single computer which makes movement decisions for all search agents. Previous research has established that distributed algorithms are preferable to centralized ones in terms of the communications and data storage

Katharina Ley is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

Kevin Comer is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

Julian Counihan is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

David Cazarnecki is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

Sarah Foster is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

Huma Husain is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

Rebecca McCrabb is a fourth-year student in the department of Systems and Information Engineering at The University of Virginia.

bandwidth needed for successful implementation [2].

The idea for an ant-based distributed search algorithm was inspired by the work of two scientists at the Université Libre de Bruxelles, Di Caro and Dorigo. They have been working on networking problems modeled after the behavior of ant colonies. Ants work together to find the shortest path from their nest to a supply of food. Each ant ventures away from the nest, leaving a chemical trail detailing that they were there and whether or not food was found. The communication between ants is indirect and asynchronous, since ants pick up their predecessors pheromone trails long after they have already passed [3]. This type of communication is called stigmergy, and is common among social insects. This type of communication is an improvement over traditional swarm-based communication since each ant, or agent, makes its own decisions about how to move instead of following the rest of the swarm. Stigmergy is also beneficial because it is easily scalable, meaning that it works regardless of how large a swarm of ants is present, as well as reconfigurable, meaning that a particular ant can exit or leave the system without damaging the overall communication network [2]. Di Caro and Dorigo's original paper looks to apply this theory of stigmergy to routing problems in computer networks. However, they believe that this type of theory could be applied to any problem that is stochastic in nature and has multiple objectives and multiple constraints.

The results of Garcia et al. (2005), though very promising, are all based on simulation data. The work of this Capstone Project provides the next step of allowing for physical testing of the ant-based algorithm. The results gained from this type of real-world demonstration of the ant-based distributed search algorithm is a vital step in assessing whether this type of algorithm can be applied to actual search problems.

IV. TEST BED IMPLEMENTATION

A. Physical Setup

The test-bed constructed for this project is located at the Engineering Special Projects Building at the University of Virginia. The actual search space consists of a level white plywood grid and a simple RGB web camera which can overlook a search space of approximately 8 x 12 feet. For searching purposes, the search space is divided into smaller search squares, each representing an area that is either a target or not a target. The search agents are Lego Mindstorms NXT robots, which are commercially available but have been modified for this project. Each robot is equipped with a constantly lit blue LED as well as a red LED which can be turned on and off remotely. This setup is pictured in FIGURE 1.

This design was chosen because it allows for the simplest type of location methodology, namely a camera-based methodology, to be employed to find the location of search agents. Additionally, the use of LED lights yields a more robust test-bed, since results are not strongly affected by lighting conditions.

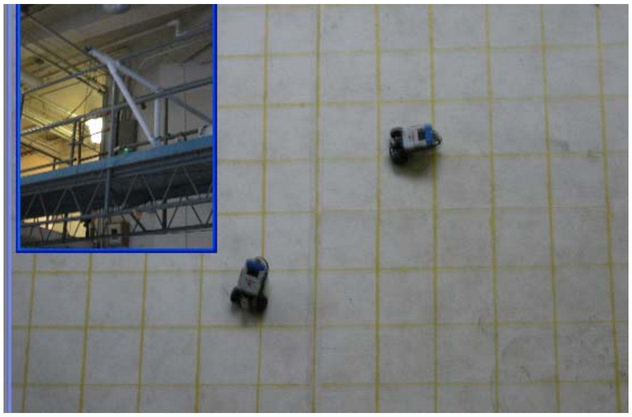

FIGURE 1: Physical Setup (Camera and Search Space)

B. Location

The most challenging aspect of this project was finding a relatively simple yet reliable method for determining the location and orientation of each search agent. In order to complete the search, robots must be guided to a precise location in the search area. Several alternatives were considered, but no commercially available products within reasonable cost range provided the accuracy needed at this small scale.

For this reason, a student-developed, camera-based approach was developed for this test-bed, using the commercially available software package MatLab. In this approach, each robot is located through its red and blue LED lights, which are fixed to the front and back of the robot as shown in FIGURE 2. Each robot, in turn, is prompted by a central computer to turn on its red LED. A camera image is then taken and analyzed to find the brightest red point using MatLab's image processing capabilities. Once this point is located, the program repeats its analysis to find the brightest blue point within a given radius of the red light already identified. These two points are used to calculate a location and orientation vector for each robot, which are also shown below. From this, the turning angle and distance required to reach a given target can be calculated.

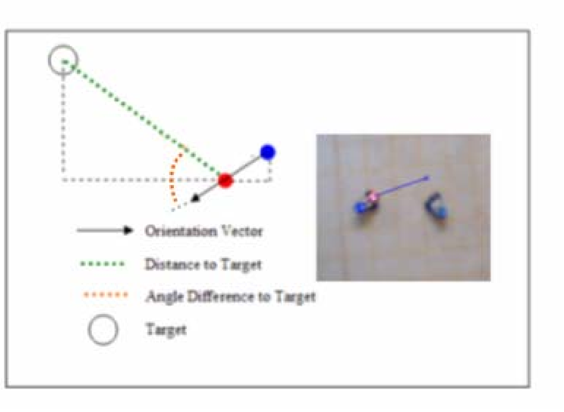

FIGURE 2: Location Geometry

C. Movement

Robot movement is controlled through a two-step process. In the first step, the angle difference between the robots orientation vector and target vector are reduced through turning. The robot can turn either right or left, depending on which direction of turn yields a smaller angle difference. Once the robot's orientation is satisfactory, meaning that the angle difference is less than 10 degrees, robots move forward in small spurts until the target location is reached. After each movement, a new location is calculated according to the process described in Section B, ensuring that the robot is on track. Target locations are calculated according to the algorithm described below

D. Target Detection

The targets used for this demonstration consist of black and white squares placed within the search area. Squares where more than 50% of the surface area is black are considered "targets," while others are considered "empty." Some empty squares do have a significant percentage of black coloring, however, thus ensuring that false positive and false negative probe results are possible. Each search agents is equipped with a light sensor directed at the ground below it, allowing it to sense for black and white coloring whenever it reaches a search location.

E. Camera Calibration

Using an overhead camera for calibration leads to image distortion in several different dimensions. In order to account for this distortion, a relationship must be established between two different planes: actual physical test-bed with metric dimensions and the camera image plane with pixel dimensions. Each point on the physical plane corresponds to a point on the image plane; this principle is called co-linearity. However, the image and actual points may be skewed, rotated, radially distorted, tangentially distorted by the camera's orientation, or randomly distorted by errors during the image capture process.

A calibration device should provide the maximum number of possible reference points in order to produce the best conversion model. The device can achieve this by splitting the search space into a large number of individual squares or by simply using a larger search space. The most effective calibration devices cover the entire span of the camera's vision. In the case of this test bed, the camera's resolution dictates that search squares must be greater than 2 x 2 inches in size. The calibration device created for this project uses a checkerboard pattern that generates over 128 pairs of physical/image coordinate points from which to create a mapping between the actual and virtual planes.

This mapping is achieved through the use of an intermediary homography matrix. This homography matrix can be determined through simple matrix mathematics using the 128 pairs of physical coordinates and image coordinates. For this project, the Gold Standard algorithm was used to calculate the intermediary homography matrix. Specifically, the Levenberg-Marquardt method was used because of its compatibility with MatLab [6]. The intermediary homography matrix yielded by this method is applied throughout the movement and search phases in order to convert pixel locations to actual locations on the search grid.

III. ALGORITHM

A. Search Process

As stated above, the search algorithm used in this project is based on the movement of ants. In such a search, the ants' pheromone trails are replaced by two probability "trails" left by each search robot. One "trail," λ, indicates the frequency with which a certain search square has been probed, and the second, μ, stores information about how likely it is that a target exists in a certain location. These two pieces of information are stored at each search square and are updated according to Bayesian theory each time an agent passes over the space.

For example, we assume that N agents are searching for n targets in an n by n grid. There are thus X = {1, 2,…,n}x{1, 2,…,n} possible target locations. As each of the N agents moves across the field, it probes its current location, obtaining a probe result Z(x) where x ∈ X. Z(x) is one if a positive probe result is returned, meaning that a target is detected, and zero otherwise. According to this probe, the posterior probability trail, μ(x), of the current search space is updated according to

$$\mu^{t+1}(x) = \frac{(1-\beta)\mu^t(x)}{\alpha(1-\mu^t(x)) + (1-\beta)\mu^t(x)}$$

if $Z_t = 1$ (1)

$$\mu^{t+1}(x) = \frac{\beta\mu^t(x)}{(1-\alpha)(1-\mu^t(x)) + \beta\mu^t(x)}$$

if $Z_t = 0$ (2)

where α is the probability of a false positive probe result, β is the probability of a false negative, and t is the iteration number. Additionally, the frequency λ(x), representing the frequency with which space x has been probed, is updated according to

$$\lambda^{t+1}(x) = \lambda^t(x) + \frac{1}{t+1}(Y - \lambda^t(x))$$

(3)

where Y is 1 if location x is probed and 0 if it is not probed. FIGURE 3 provides sample output graphs of a MatLab simulation of this type of search. These graphs show how probing frequency and probability of target existence relate to actual target locations, represented by red circles.

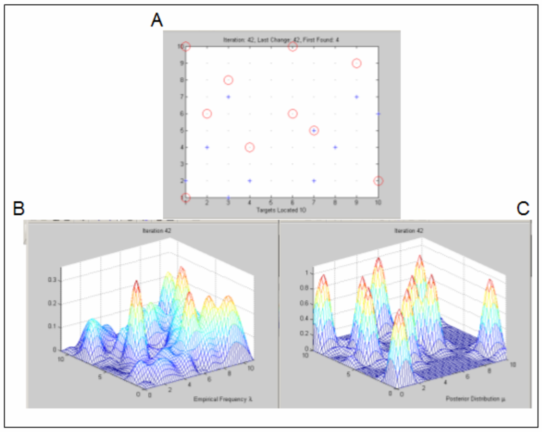

FIGURE 3: Simulation Output Graphs (10 agents, 10 targets) (A) simulation of search, (B) frequency trail λ, (C) probability trail μ

The final step of the algorithm consists of deciding which space to search next. Each agent N must pick one of the 8 squares adjacent to it, which we define as being part of the set X_N. This decision is made by choosing a square that has a high probability of containing a target (μ(x) is close to one), and a low frequency of having been probed previously (low λ(x)). These two considerations ensure fast location of targets while avoiding collision among robots. Mathematically, the decision of where to search next can be expressed as minimizing the expected value of the reward function, $U_N(x)$ of robot N probing square x between the 8 adjacent squares:

$$s_N^{t+1} \in X_N \quad \text{where}$$

$$E[U_N(s_N^{t+1})] = \max_{x \in X_N}[r\mu^t(x) - h\lambda^t_{-N}(x)]$$

(4)

Here s_N^{i+1} is the next probe location of robot N, r is the reward associated with successful target detection, h is the penalty for duplicating the probe of another robot, and $\lambda^t_{-N}(x)$ is the frequency of probes performed at location x by robots other than N [2]. For purposes of this testbed the location of the square that should be searched next is passed to the agent in the form of coordinates of the center of that search square.

Preliminary results gathered from simulations of this type of ant-based are very promising. When compared to a centralized scheme, the swarm-based distributed algorithm performs almost equally well when compared on the basis of number of iterations required to detect all targets [4]. Additionally, the relative performance of the distributed algorithm seems to improve as the number of agents and targets as well as the size of the search space improve.

B. Algorithm Implementation

Due to processing limitations in the Lego Mindstorms NXT robots, the majority of calculations required for this algorithm are performed by a central computer and have been implemented using MatLab. The decentralized nature of the system is preserved, however, since the decision about where to probe next is made independently for each robot, as though the robot itself were making the decision without any communication with other agents. False positive and false negative probe results are artificially computer generated according to a user-determined likelihood of these events.

III. RESULTS

The main goal of this project was to provide a test-bed for swarm-based distributed search. This goal has been accomplished, even though major obstacles had to be overcome to do so. Challenges arose in early stages of the project, and especially the implementation of an accurate robot location system proved to be much more difficult than was originally projected. Due to the amount of accuracy required and the changeability of lighting conditions in the work space, robot location is still not entirely reliable and should be improved before this test-bed is used for testing of other algorithms. Significant headway was made in this area, yielding a location system that provides accurate locations in approximately 90% of cases. However, a non-camera-based system may need to be used in order to improve this performance further.

Additionally, moving robots to a specified target is still a somewhat time-consuming process, since small movements must be made and multiple locations calculated before a robot is in the correct position to conduct its next probe. Due to slip of the robot wheels, robots do not always respond precisely to movement commands, making it necessary to re-check their location after each movement is made. Several different movement surfaces and movement schemes were tested, yet it is still not possible to move a robot to a specific location in only one iteration.

The implementation of the actual search algorithm was very successful. Although most calculations are performed by a centralized computer, the distributed nature of the algorithm is preserved by limiting information exchange between robots. The algorithm implemented here functions in the same way as previous computer-based simulations, with the exception that computer-generated search probes have been replaced by actual light sensor readings.

Despite some shortcomings, a working prototype of a distributed search test-bed has been created through the work of this project. Four robots are able to successfully navigate a search space, making independent movement decisions and successfully locating targets. Due to occasional failures in the locating and moving of robots as well as the detecting of targets, not every iteration of robot movement leads to a probability and frequency update, thus meaning that the physical test-bed requires more iterations to achieve the same results as its computer-based counterpart. Nevertheless, the overall behavior of the physical simulation closely mirrors the previously observed performance of the computer simulation, thus indicating that the test-bed functions correctly.

IV. FUTURE RESEARCH

Several steps should be taken in order to further improve upon and increase the benefits of the project work performed here. Firstly, improvements on the location and movement procedures can still be made. A system for search tracking should be implemented so that a human user may better understand how the search proceeds. Secondly, this test-bed could be adapted to test other types of algorithms, both distributed and centralized, that have previously been studied only through computerized simulation. Thirdly, additional modifications to the current algorithm could be made and tested. Several such modifications, such as tools for recognizing patterns within the target distribution of a given search space, are already under study.

A final area of future work would be to more closely study the scalability and efficiency of the algorithm as it functions in the test-bed designed here. For example, larger and smaller searches should be conducted, and results should be compared to the performance of other search algorithms attained using the same test-bed.

V. CONCLUSION

The main conclusion that can be drawn from this project is that it is possible to construct a small-scale distributed search test-bed within a reasonable budget. While preliminary research for this project suggested that even the development of a location program for this type of test-bed may be too difficult and costly for the project team, such a system is now functioning with about 95% consistency. This location system has made it possible to overcome some of the shortcoming of the Lego Mindstorms NXT robots to move accurately through repeated location checks, thus creating a system where multiple robots can be reliably sent to multiple specific locations on command. This is the true basis of the distributed search test-bed, since the actual implementation of the specific algorithm is merely an adaptation of existing computer simulation code.

ACKNOWLEDGEMENTS

Special thanks to Professors Alfredo Garcia and Peter Beling of the University of Virginia Department of Systems and Information Engineering for their work as Capstone project advisors. Additional thanks go out to Chenyang Li and Fernan Pedraza, both graduate students in the Department of Systems and Information Engineering, for their help and support with the thesis project and paper throughout the past year. The use of resources provided by the University of Virginia Wireless Internet Center for Advanced Technology (WICAT) is also greatly appreciated.

REFERENCES

[1] Beling P. Flanagan, M., Noakes, P., Paulsen, L., Verley, B. and Weightman, J. 2006. "Decentralized Coordination Processes", Proceedings of 2006 IEEE Systems and Information Engineering Design Symposium.

[2] Beling P., Beaulieu B., Durham I., McKinstrie R., Shumate P., Stamper K., Verell E. 2005. "Dynamic Multi-agent Coordination: RoboCops", Proceedings of 2005 IEEE Systems and Information Engineering Design Symposium.

[3] Di Caro, G., Dorigo, M., 1998 "AntNet: Distributed Stigmergic Control for Communications Networks," Journal of Artificial Intelligence Research 9, 317-365

[4] Garcia A. Li Ch and Pedraza F. 2007. "Rational Swarms for Distributed On-line Bayesian Search" Working paper, available at: http://www.sys.virginia.edu/techreps/index.html

[5] Garcia, A., Campos, E. and Li, Ch., 2005 "Distributed On-line Bayesian Search," Proceedings of First Workshop on Stochasticity in Distributed Systems.

[6] Hartley, Richard.; Zisserman, Andrew. 2004. "Multiple View Geometry in Computer Vision" Cambridge University Press, New York.