A Search Algorithm for Teams of Heterogeneous Agents with Coverage Guarantees

Christopher W. Lum*, Rolf T. Rysdyk†, and Juris Vagners‡
Autonomous Flight Systems Laboratory
University of Washington, Seattle, WA, 98195, USA

Among common ISR tasks, searching for a target in a complex environment is a problem for which autonomous systems are well suited. This work considers the problem of searching for targets using a team of heterogeneous agents. The system maintains a grid-based world model which contains information about the probability that a target is located in a given cell of the map. Agents formulate control decisions for a fixed number of time steps using a modular algorithm that allows parameterizations of agent capabilities. This paper investigates a solution that guarantees total map coverage. The control law for each agent does not require explicit knowledge of other agents. This yields a system which is scalable to a large number of vehicles. The resulting search patterns guarantee an exhaustive search of the map in the sense that all cells will be searched sufficiently to ensure that the probability of a target going unnoticed is driven to zero. Modifications to this algorithm for explicit cooperation between agents is also investigated.

Nomenclature

B	Set of $\overline{z}$ values defining spatial domain of occupancy based map
B_R	Locations reachable by agent in d steps
$\tilde{B}$	Set of $\overline{z}$ values defining center of occupancy based map cells
$\tilde{B}_R$	Cell centers reachable by agent in d steps
d	Prediction horizon, number of waypoints in path
d_i	Minimum distance from generator i to set $\tilde{B}_{max}$
$f_\chi()$	Reward function for course deviation in $(\wp_2)$
$f_d()$	Reward function for distance in $(\wp_2)$
$f_h()$	Reward function for high score cell in $(\wp_2)$
h	Sensor reliability factor in range $[0, 1]$
I	Index of agent closer to $\tilde{B}_{max}$ than any other agent
$I_{\hat{n}(m)}$	Indices corresponding to runs where at least m agents find the target
$J_0()$	Total reward function for $(\wp_2)$
L_x, L_y	Width and height of map cell in x-direction and y-direction, respectively
N_x, N_y	Number of columns and rows, respectively, of occupancy based map
$\tilde{N}(i)$	Number of agents who find target in run i
$\tilde{N}_{ave}$	Average number of agents who find target
$\hat{n}(m)$	Number of runs where at least m agents find the target
P	Set of Voronoi generator points
$(\wp_1)$	Subproblem of creating future world state estimates
$(\wp_2)$	Subproblem of finding desirable cells for agent to search
$(\wp_3)$	Subproblem of finding trajectories for agent's path
$p(A\|B)$	Conditional probability of A given B

*Research Assistant, Dept. of Aeronautics and Astronautics, lum@u.washington.edu, AIAA student member
†Advanced Development, Insitu, rolf.rysdyk@insitu.com, AIAA member
‡Professor Emeritus, Dept. of Aeronautics and Astronautics, vagners@aa.washington.edu, AIAA member

p_i	Voronoi generator point i
$q(a,b)$	Calculates absolute angular difference between angles a and b
R_{max}	Maximum distance agent can travel in a d steps
$S(i,k)$	Cumulative map score at step k for run i
$S_{ave}(k)$	Average cumulative map score at step k
s_k	Score of given occupancy map cell at step k ($p(X_k = x_B)$ at step k)
$T_m(i)$	Time when the m^{th} agent finds the target for run i
$T_{m,ave}$	Average time when the m^{th} agent finds the target
$\overline{V}$	Voronoi diagram
V_{max}	Max velocity of agent
V_t	Variance of weights at time t
$V(i,k)$	Variance of map scores at step k for run i
$V_{ave}(k)$	Average variance of map scores at step k
x	Spatial coordinate
x_A, x_B	Event of target not in cell and target in cell states, respectively
x_{min}, x_{max}	Minimum and maximum x value of occupancy based map
$x_w(k,\overline{z})$	Actual state of the world at time step k and location $\overline{z}$
$\hat{x}_w(k,\overline{z})$	Estimated world state at time step k and location $\overline{z}$
y_{min}, y_{max}	Minimum and maximum y value of the occupancy based map
$\overline{z}$	(x,y) coordinate $(P_E\ P_N)^T$
$\overline{z}^\star$	Most desirable location in $(\wp_2)$
$\overline{z}_{agt}$	Agent's current (x,y) position
z_A, z_B	Observation of target not in cell and in cell, resectively
$\overline{z}_0$	Agent's current coordinate
$\overline{z}_H$	Location of cell center with highest score in map and closest to agent
$\overline{z}_h$	Location of cell center in agent's reachable set closest to $\overline{z}_H$
$\overline{z}_s$	Location of cell center of same cell that agent is currently in
α	Scalar tradeoff parameter for $\hat{x}_w(k+d,\overline{z})$
β	Scalar tradeoff parameter for $f_\chi(\overline{z})$
ΔT	Time between steps/waypoints
η	Maximum score of cell within agent's reachable set
γ	Scalar tradeoff parameter for $f_d(\overline{z})$
φ	Distance from agent to target
ζ_i	Flag used in definition of $\overline{z}_{h_i}$

I. Introduction

The overwhelming majority of current missions tasked to autonomous systems revolve around intelligence, reconnaissance, and surveillance (ISR). This work seeks to increase the autonomy and capabilities of a heterogeneous team of agents involved in a search and surveillance type mission. The system maintains a world model which includes an estimate of possible target states and the state of the environment. The issue of compelling agents to converge on possibly moving targets and continuing to search new regions is formulated as a model predictive control problem. The world model is propagated forward in time and strategic decisions are made based on the predicted future state of the world. Agents formulate control decisions for a fixed number of time steps by optimizing an objective function that allows for control and timing constraints.

The overall objective of this work is to develop robust and scalable control laws to apply to a heterogeneous group of agents so that they operate in a cooperative fashion to achieve a common goal. The goal in this application considers the specific mission of coordinating a group of agents to search a two dimensional space for a target. Although this goal may seem narrow in scope, many of the technologies developed for this application are easily adaptable to more general tasks encountered in autonomous systems. This paper investigates a modular algorithm which can efficiently coordinate a team of possibly heterogeneous agents and ensure that the team performs an exhaustive search of the map.

This type of search mission has been previously studied by several other groups. Classically, it has been

addressed as classical exploration missions by Monekosso[1] and Dollarhide[2] et al. It also fits the model of search and rescue applications as shown by Kurabayashi,[3] Kantor,[4] and Jennings[5] et al. A survey of classical searching techniques by Benkoski[6] provides additional background and references.

One modern approach that has become popular is to consider possible target locations as a continuous or discrete probability density function. Groups such as Durrant-Whyte et al.[7,8,9] studied the problem of searching for a target using a Bayesian probabilistic approach and have investigated some of the communication issues involved in such a search. Polycarpou et al.[10,11,12] applied optimization techniques to generate search patterns over a finite amount of steps. Many of these methods are successful and effective but have difficulty providing guarantees on target detection and map coverage. To address this, Erignac[13] developed exhaustive searching strategies that also provide guarantees about map coverage with ideas based on pheromone maps. Coverage of maps and domains have also been studied in the context of minimum service time to spontaneously occurring targets. Many of these techniques use Voronoi partitioning of the domain to maintain agent separation and coverage. This approach has been studied by Du,[14] Cortes[15,16] , and Frazzoli[17,18] . Searching and map building has been studied extensively by the robotics community as well. Groups such as Fox et al.[19] have looked at generating searching algorithms with the ideas of exploration and map building in mind. Others such as Baillieul et al.[20] and Hoffman[21] have posed the searching mission in an information theoretic framework which provides useful metrics for search effectiveness. Previous work at the University of Washington by Rubio et al.[22] investigated searching algorithms in the context of adaptive algorithms. Previous work[23] explored the use of novel optimization techniques to address the search problem.

These methods, while effective for their respective applications, contain shortcomings when addressing the search problem considered in this application. To address the problem at hand, a modular search algorithm is presented in this paper. The algorithm is modular in the sense that it is comprised of three main steps. Each step is somewhat independent of the others and different algorithms can be used to accomplish the same goal within any of the three main steps.

The first step involves propagating the world state forward in time. By providing a predictive aspect to the problem, each agent can then make control decisions based on the predicted future state of the world rather than only using the current information. Once the system generates a predicted future world state, each agent determines a desirable coordinate to visit in the future by solving a numerical optimization problem. This formulation allows for each agent in the team to have a different set of parameters; therefore, each agent in the team can have its own notion of desirability. Finally, once this desirable coordinate is determined, a path that transitions the agent from its current location to the desirable coordinate can be applied.

The focus of this paper is an algorithm for solving the second step (desirable location selection) of the process which will guarantee that the team exhaustively searches the map for the target. The first[23] (generating a predictive world estimate) and third[24] (path planning) steps have already been addressed in previous works.

Section II describes the notion of occupancy based maps and their features. These maps provide the framework for the search strategy which is initially derived as a single agent search strategy and is described in Section III. Section IV builds on these ideas and introduces modifications to allow for explicit cooperation. Simulation results and performance metrics are discussed in Section V. Finally, Section VI presents conclusions and Section VII illustrates some future directions of research.

II. Occupancy Based Maps

In order to effectively search a two dimensional domain for a target, the system must keep track of the state of the world in terms of possible target locations. To do this, an occupancy based map is employed. These constructs were originally developed by Elfs[25,26] but functionality such as Bayesian score updates and time varying models are added to the maps to accommodate the algorithm.

A. Defining Occupancy Based Maps

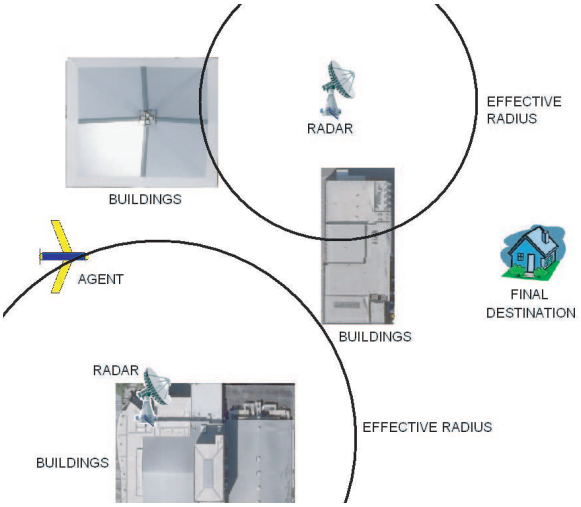
The search domain is discretized into rectangular cells. Each cell is assigned a score which is the probability that the target is located in that cell. This is similar to a two dimensional, discretized probability density function.[7] The spatial domain of the occupancy based map consists of a box where x is between x_{min} and

x_{max}. Similarly for the y dimension.

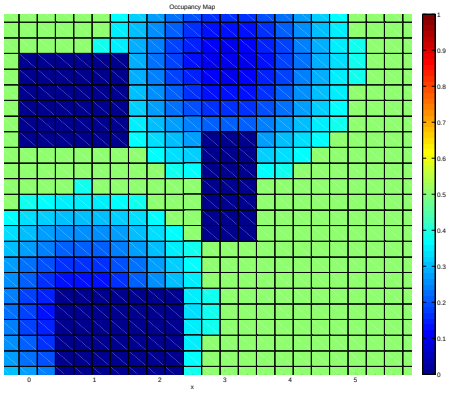
$$B = \left\{ \overline{z} = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \mid z_1 \in [x_{min}, x_{max}], z_2 \in [y_{min}, y_{max}] \right\} \tag{1}$$

The occupancy based map, $x_w()$, is a function defined over the set $B \times \Re$ which assigns a score in the range $[0, 1]$ to each element $\overline{z} \in B \subset \Re^2$ at a certain time step $k \in \Re$. In other words, $x_w : B \times \Re \rightarrow \Re$. The score of a given cell represents the probability that the target is located in that cell.

The occupancy based map is shared and updated by all agents involved in the search. At each time step, guidance decisions for each agent are computed based on this map. The state of the map at any time k is also referred to as the world state. This reflects the fact that the map represents the possible locations of targets and other objects in the environment. In essence, the system's belief of the state of the world is embedded in the state of the occupancy based map. For example, the map can represent the locations of obstacles and reward areas in the environment. The idea of embedding obstacles in the map is similar to defining obstacles in a configuration space.[27] An example of embedded obstacles and reward areas in an occupancy based map is shown in Figure 1.



(a) Physical environment with soft and hard obstacles



(b) Occupancy based map representation of environment

Figure 1. Abstraction of urban environment using occupancy based maps.

In this example, the dark blue sections represent cells with zero scores (hard obstacles) and the green sections represent scores of 0.5 (neutral values). Furthermore, there are regions which correspond to soft obstacles that should be avoided if possible but entering these regions does not violate a constraint. These sections are represented by the lighter blue shades with scores ranging from 0 to 0.5. If there were regions that were beneficial to the agent, these could be assigned scores greater than 0.5.

It is useful to define the cell center set, $\tilde{B}$, as all values $\overline{z}$ which correspond to the center of a cell in the occupancy based map.

$$\tilde{B} = \left\{ \overline{z} \middle| \overline{z} = \begin{pmatrix} x_{min} + L_x(i - 1/2) \\ y_{min} + L_y(j - 1/2) \end{pmatrix}, i = 1, 2, \ldots, N_x, j = 1, 2, \ldots, N_y \right\} \tag{2}$$

B. Updating Map Scores

The occupancy based map is dynamic and can be updated either by agents involved in the mission, external sources, or other means. The agents are able to modify the map to reflect their findings during the search mission. Each agent in the team is able to search the cell at its current location using its sensor. The discrete state space of the cell is simply $X_k = \{x_A, x_B\}$ where $X_k = x_A$ corresponds to the target not in the cell and $X_k = x_B$ meaning that the target is in the cell. In a similar fashion, the agent may make one of two sensor measurements, $Z_t = z_A$ (observe target not in cell) and $Z_t = z_B$ (observe target in cell). As mentioned previously, the score of a given cell in the occupancy based map reflects the scalar probability that the target

is located in that cell at the current time step k. For convenience, the score of the cell at time step k is denoted $s_k = p(X_k = x_B)$.

To model a heterogeneous team of agents with stochastic sensors, each agent's sensor is assigned a reliability factor $h \in [0, 1]$. A value of $h = 0$ implies that the sensor is completely unreliable and no information can be gained from this sensor. Conversely, $h = 1$ corresponds to a completely reliable sensor that can ascertain if the target is or is not located in the agent's current cell in a single measurement. The probabilistic sensor model can be formed as

$$\begin{aligned} p(Z_t = z_A | X_t = x_B) &= 1 - \tfrac{1}{2}(h+1) \\ p(Z_t = z_B | X_t = x_B) &= \tfrac{1}{2}(h+1) \end{aligned} \tag{3}$$

Assuming that the state of any given occupancy map cell score is not affected by the action of taking a measurement, the probabilistic score of a given occupancy based map cell can be updated using the sensor model of Eq. 3 which yields the following Bayesian update rule.

$$s_k = \begin{cases} \frac{s_{k-1}(1-h)}{1+(1-2s_{k-1})h} & \text{if } Z_k = z_A \\ \frac{s_{k-1}(1+h)}{1+(2s_{k-1}-1)h} & \text{if } Z_k = z_B \end{cases} \tag{4}$$

In Eq. 4, s_k represents the score of the occupancy map cell ($s_k = x_w(k, \overline{z})$ for $\overline{z}$ in some cell region).

This update rule has several interesting properties. It can be shown that the scores of each cell either monotonically increase or decrease with each sensor measurement for the majority of values of h and s_{k-1}. It is trivial to see that if $s_{k-1} = 1$ then $s_k = 1$ and if $s_{k-1} = 0$, then $s_k = 0$ for either case of $Z_k = z_A$ or $Z_k = z_B$ and for all values of h. The physical significance of this is that if the target is absolutely certain to either be in the cell or not, then no further updates will change this fact. Similarly, for $Z_k = z_A$, if $h = 0$ then $s_k = s_{k-1}$ and if $h = 1$ then $s_k = 0$. This implies that if the sensor is completely unreliable ($h = 0$) and making a measurement with this sensor will not change the state of the cell. Conversely, if the sensor is completely reliable with $h = 1$, then only a single measurement of $Z_k = z_A$ is all that is required to change the score of the cell to 0. A similar situation arises for $Z_k = z_B$ where if $h = 0$ then $s_k = s_{k-1}$ but if $h = 1$ then $s_k = 1$. This is because z_B with $h = 1$ is the event of detecting the target with a 100% reliable sensor. The more interesting cases are when s_{k-1} and h are in the interval $(0, 1)$.

Theorem II.1. *Under the update rule of Eq. 4, $s_k < s_{k-1}$ if $Z_k = z_A$ and $s_k > s_{k-1}$ if $Z_k = z_B$ $\forall s_{k-1}, h \in (0, 1)$.*

Proof. For the case of $Z_k = z_A$ the burden is to now show that $s_k < s_{k-1}\, \forall k$ for the cases where $s_{k-1}, h \in (0, 1)$.

One can examine the denominator of Eq. 4 and note that $\forall s_{k-1} \in (0, 1)$ the term $1 - 2s_{k-1} > -1$. Therefore, the denominator term must be greater than $1 - h$. Noting that for all $h \in (0, 1)$ the term $1 - h > 0$. Therefore, the denominator term must always be positive.

Furthermore, it is easy to see that $1 - s_{k-1} > 0$ and $2h > 0$ for these conditions. So one can write

$$0 < 2h(1 - s_{k-1}) \tag{5}$$

$$1 - h < 1 + h - 2hs_{k-1} \tag{6}$$

$$\frac{1-h}{1+h-2hs_{k-1}} < 1 \tag{7}$$

$$\tag{8}$$

Multiplying both sides by s_{k-1} yields the final result

$$\frac{s_{k-1}(1-h)}{1+(1-2s_{k-1})h} = s_k < s_{k-1} \;\; \forall h, s_{k-1} \in (0, 1) \tag{9}$$

It is trivial to show that $s_k > 0\, \forall k$. This in conjunction with Eq. 9 shows that with enough sensor measurements of $Z_k = z_A$, the score of a given cell will proceed monotonically towards 0.

A similar proof can be applied for the case of $Z_k = z_B$ with the result showing that with enough sensor measurements of $Z_k = z_B$, the score of a given cell will proceed monotonically towards 1. □

A standard Markov assumption is used to remove the dependence of the next score on past scores, and therefore only one version of the occupancy based map must be maintained at any given time step. The map can be made time varying using a number of techniques.[28]

One final aspect of system modeling that concerns updating the map is how the agents update the map when they encounter a target. If the agent searches the cell where the target is located and successfully finds the target, it can update the scores of its current cell and those around it. The sensor reliability factor is also used to model a sensor which may miss a positive target identification. If h is low, there is a high probability that the agent will not find the target even if it searches the correct cell. This behavior affects several performance metrics such as the average time to target detection, which is discussed later in Section V.

The occupancy based map and its associated features provides a versatile framework from which to build a searching algorithm.

III. Single Agent Search Strategy

The searching strategy is addressed in this section. The problem of searching an area using a team of possibly heterogeneous, autonomous agents is still an open problem. Many of the sources listed in Section I provide methods to perform this mission with varying degrees of success. For example, randomized coverage methods[17,16] are unable to provide guarantees regarding map coverage and target detection. Other approaches such as Bayesian searching[7,8,9] have difficulties embedding information about a complex environment into the algorithm. This section focuses on addressing these issues and developing a modular and scalar algorithm[29] that can be applied to this situation.

A. Algorithm Overview

It is desired that agents in the team exhibit certain behaviors and achieve several goals during a mission. One desired behavior is that the agent is to converge on regions of high score (a high probability that the target is located in a given location). Once an agent locates an anomaly, it should loiter there until a positive identification[30] can be made. While this occurs, other agents who are farther away will continue searching. A search strategy for a single agent is developed with these goals in mind and then applied to each agent in the team. The overall flow of the search strategy for a single agent is shown in Figure 2.

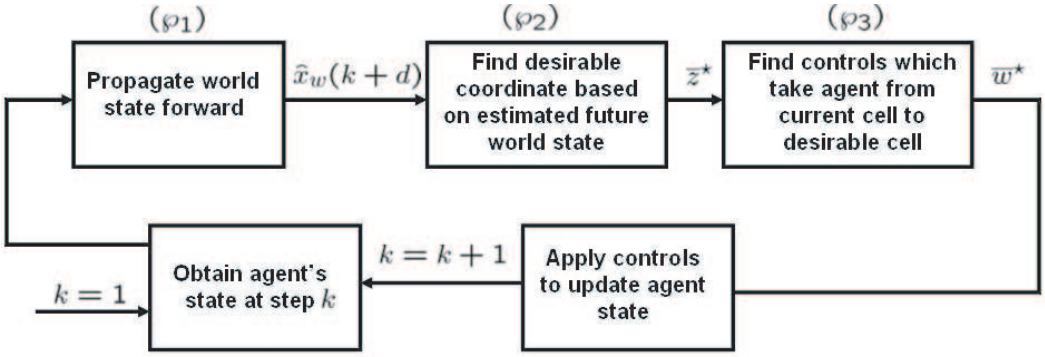


Figure 2. Flow diagram for single agent search strategy.

The single agent search strategy is comprised of three subproblems which are referred to as $(\wp_1)$, $(\wp_2)$, and $(\wp_3)$. The process starts by finding the agent's state at the current time, $\overline{x}_{agt}(k)$. Next, $(\wp_1)$ is solved to obtain the estimated world state at time $k+d$. Next, $(\wp_2)$ is solved to find a desirable coordinate, $\overline{z}^\star$, which the agent will visit within the next d steps. Finally, $(\wp_3)$ consists of finding a set of waypoints/controls, $\overline{w}^\star$, which will take the agent from its current location to the location of the desirable coordinate found in $(\wp_2)$. The modular nature of the algorithm benefits analysis since each sub-problem can be analyzed independently. This paper focuses on a solution for sub-problem $(\wp_2)$ and assumes that a predictive algorithm[31] is available to solve $(\wp_1)$ and a path planning algorithm[24] exists to solve $(\wp_3)$.

Each agent in the team follows this same policy without information or explicit knowledge of other members of the team. This approach has several advantages and disadvantages. The major benefit of this strategy is that the computational resources needed for this algorithm grow linearly with the number

of agents. Each agent does not require explicit knowledge of other agents to compute its own control law. This allows the algorithm to be scalable to a large number of agents. Furthermore, although the algorithm is centralized in the sense that there must be a star communication topology where each agent is in communication with a centralized base, the computation for the algorithm can be performed in a decentralized manner in the sense that each agent can make its own decisions based on information from the base without consulting other agents in the team.

The primary disadvantage of this policy is that there is no explicit cooperation between agents. As shown in many situations, explicit cooperation has the potential to improve performance of the system at the cost of increased complexity and computational resources[32,33,9] . Some modifications to allow for explicit cooperation are presented in Section IV.

B. ($\wp_2$) Desirable Location Selection

The agent needs to choose a location that it must travel to based on the estimated future state of the world. This section describes a method that can be used to choose a desirable cell to search. In ($\wp_1$), the state of the world is propagated forward in time by d steps. Subproblem ($\wp_2$) concerns finding a desirable location (a desirable $\overline{z}$ value) for the agent to travel to in d steps. The desirability of a location is determined by a reward function of the following form

$$J_0(\overline{z}) = \alpha \hat{x}_w(k+d, \overline{z}) + \eta\left(\beta f_\chi(\overline{z}) + \gamma f_d(\overline{z})\right) + \delta f_h(\overline{z}) \tag{10}$$

The reward function is a combination of terms which model both the environment and agent states. For example, the term $\hat{x}_w(k+d, \overline{z})$ measures the estimated state of the world d steps into the future. The function $f_\chi()$ is a function which penalizes heading/course angle changes. In a similar fashion, $f_d()$ rewards locations which are farther away from the agent. Finally, $f_h()$ is an indicator function which serves to drive the agent towards the highest score in the map. This describes the general behavior of the cost function, but a more detailed look at its construction is necessary.

With the cost function defined, the most desirable location is then found via an optimization scheme using Eq. 10 as an objective function. Typically, the set over which one optimizes Eq. 10 is the set of all locations that the agent can reach in d steps. This set is determined by parameters such as its maximum velocity and time step. The set of all locations that the agent can reach in d steps is referred to as the agent's reachable set, $B_R \subseteq B$. This application defines B_R as

$$B_R = \{\overline{z} \in B \mid ||\overline{z} - \overline{z}_{agt}|| \leq R_{max}\} \tag{11}$$

Eq. 11 assumes that the agent has no turn rate limits and the agent has simple planar kinematics. In a practical application where there may be saturation concerns, it is possible that B_R may not be a perfect circle as described in Eq. 11. In this case, it simply becomes more difficult to define and compute B_R but the following analysis is not affected by the geometry of B_R.

It is useful to also define the set of cell centers that the agent can reach in d steps. This is simply

$$\tilde{B}_R = B_R \bigcap \tilde{B} \tag{12}$$

In Eq. 10, the function $f_\chi()$ is given by

$$f_\chi(\overline{z}) = \begin{cases} 0 & \text{if } \overline{z} \text{ in same cell as current agent} \\ 1 - \frac{q\left(\chi_{agt}, \pi/2 - \text{atan2}(\overline{z}_2 - y_{agt}, \overline{z}_1 - x_{agt})\right)}{\pi} & \text{otherwise} \end{cases} \tag{13}$$

In Eq. 13, the function $q(a,b)$ computes the absolute angular difference between the two angles, a and b. A simple absolute value of the difference of a and b is not sufficient and some simple heuristics are included in the function $\eta()$ of Eq. 10 to take care of situations such as where $a = 1 \cdot \pi/180$ radians and $b = 359 \cdot \pi/180$ radians. A simple absolute value of the difference would return an angle of $358 \cdot \pi/180$ radians, which is incorrect. However, the function $\eta()$ returns the correct angular difference of $2 \cdot \pi/180$ radians. Note that the range of $f_\chi()$ is $[0, 1]$. An example of this function is shown in Figure 4(c).

The function $f_d()$ of Eq. 10 is given by

$$f_d(\overline{z}) = \begin{cases} \frac{||\overline{z} - \overline{z}_{agt}||}{R_{max}} & \text{if } \overline{z} \in B_R \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The function $f_d()$ effectively rewards locations that are farther away from the current agent position until the distance R_{max} is reached; past this point, the function returns 0. An example of this function is shown in Figure 4(d).

Finally, the function $f_h()$ of Eq. 10 is an indicator function. To define this function, it becomes necessary to first define some intermediate variables. The first of these variables is given by

$$\tilde{B}_{max} = \left\{ \overline{z} \in \tilde{B} \mid \hat{x}_w(k+d, \overline{z}) \text{ is maximum } \forall \overline{z} \in \tilde{B} \right\} \tag{15}$$

The set $\tilde{B}_{max}$ is simply the set of cell centers that have the highest score in the map. Note that this set may have more than one cell center. For further clarification, the relationship between the sets $\tilde{B}_{max}$, B, B_R, and $\tilde{B}_R$ are shown later in Section IV, Figure 7.

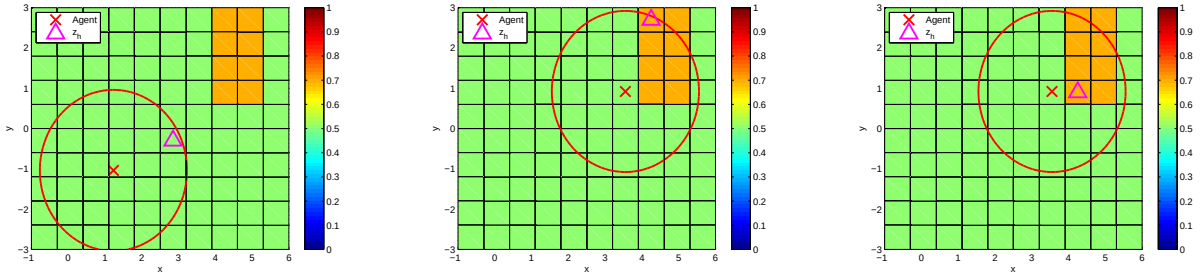
A single cell center from $\tilde{B}_{max}$ can be chosen using one of two methods. The first method involves choosing the point in $\tilde{B}_{max}$ which is closest to the agent. This location is designated as $\overline{z}_H$

$$\overline{z}_H \in \arg \underset{\overline{z} \in \tilde{B}_{max}}{\text{minimize}} \; ||\overline{z} - \overline{z}_{agt}|| \tag{16}$$

An alternative method for choosing $\overline{z}_H$ is to use

$$\overline{z}_H \in \begin{cases} \arg \underset{\overline{z} \in \tilde{B}_{max}}{\text{minimize}} \; ||\overline{z} - \overline{z}_{agt}|| & \text{if } \tilde{B}_{max} \bigcap \tilde{B}_R = \emptyset \\ \arg \underset{\overline{z} \in \tilde{B}_{max} \bigcap \tilde{B}_R}{\text{maximize}} \; ||\overline{z} - \overline{z}_{agt}|| & \text{otherwise} \end{cases} \tag{17}$$

The method for choosing $\overline{z}_H$ in Eq. 17 entails first checking if the set $\tilde{B}_{max} \bigcap \tilde{B}_R = \emptyset$. If this is true, this implies that there are no cells that have the highest score of the map in the agent's reachable set. If this is the case, Eq. 16 and Eq. 17 will choose the same point for $\overline{z}_H$. This can be seen in Figure 3(a). The two methods deviate if there is a cell with the highest score of the map within the agent's reachable set. In this case, Eq. 16 would still choose the cell in $\tilde{B}_{max}$ which is closest to the agent as shown in Figure 3(c). On the other hand, Eq. 17 would instead choose the cell with the maximum score that is farthest from the agent, but is still in the agent's reachable set as shown in Figure 3(b). Either method can be used and each has advantages and disadvantages that will be discussed in Section IV.



(a) $\overline{z}_h$ using either Eq. 16 or 17 (b) $\overline{z}_h$ using Eq. 17 (c) $\overline{z}_h$ using Eq. 16

Figure 3. Differences between using Eq. 16 and 17 for choosing $\overline{z}_H$.

With the point $\overline{z}_H$ defined, the point in the agent's reachable cell centers that is closest to $\overline{z}_H$ is given by

$$\overline{z}_h \in \arg \underset{\overline{z} \in \tilde{B}_R}{\text{minimize}} \; ||\overline{z} - \overline{z}_H|| \tag{18}$$

With $\overline{z}_H$ and $\overline{z}_h$ defined, the function $f_h()$ is simply given as

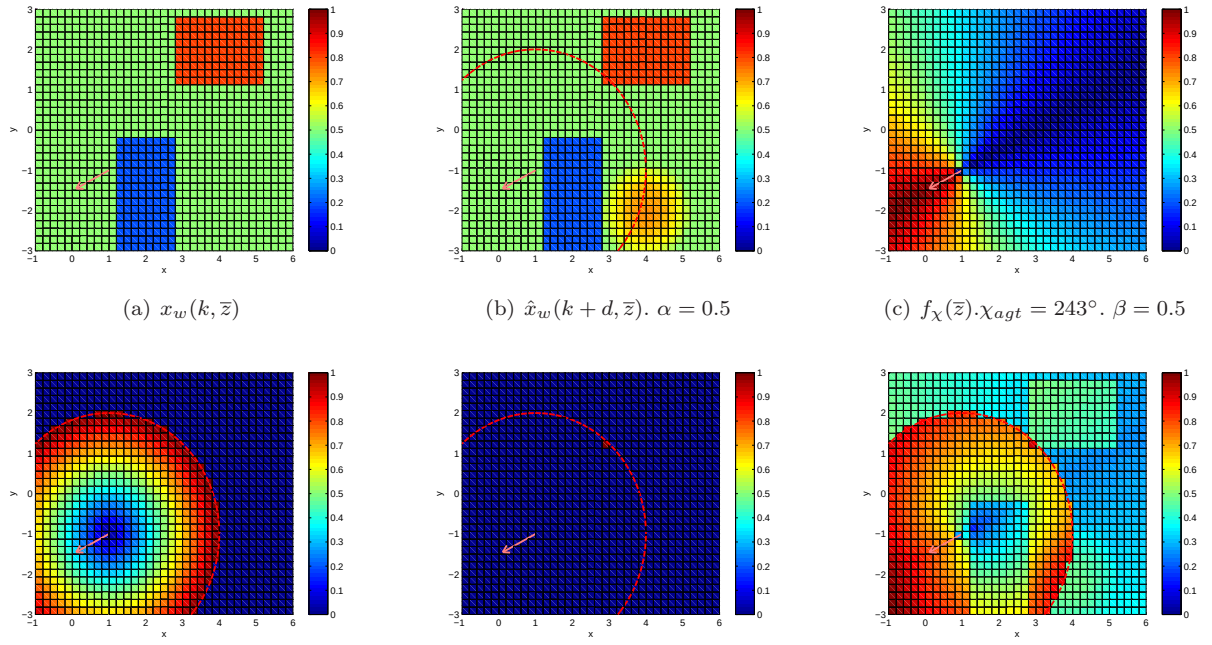
$$f_h(\overline{z}) = \begin{cases} 1 & \text{if } \overline{z} \text{ is in cell containing } \overline{z}_h \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

An example of the $f_h()$ function is shown in Figure 4(e).

The final parameter in the reward function is the variable η which is the maximum score within the agent's reachable set.

$$\eta = \max_{\overline{z} \in \tilde{B}_R} \hat{x}_w(k+d, \overline{z}) \tag{20}$$

An example of the various functions that make up the reward function are shown below in Figure 4.



(a) $x_w(k, \overline{z})$ (b) $\hat{x}_w(k+d, \overline{z})$. $\alpha = 0.5$ (c) $f_\chi(\overline{z})$.$\chi_{agt} = 243°$. $\beta = 0.5$

(d) $f_d(\overline{z})$. $\gamma = 0.5$ (e) $f_h(\overline{z})$. $\delta = 0.05$ (f) $J_0(\overline{z})$

Figure 4. Various functions which compose the reward function, $J_0()$.

Figure 4(a) shows the state of the world at the current time. This is the system's belief of the world without making any estimates of the target location.

Figure 4(b) shows the estimated state of the world in d steps. In this case, an external source has provided information that the target is likely located in the lower right corner. This estimate can be made by a team member or another source in the system. The dashed red line shows the maximum distance the agent can reach in d steps. Maximizing this function over $\tilde{B}_R$ requires choosing the cell with the highest score within the dashed red line.

Figure 4(c) shows the $f_\chi()$ function for the particular case of the agent having a course angle of 243°. Maximizing this function requires picking a location $\overline{z}$ which minimizes the course change required to visit this location ($\overline{z}$ locations which are in line with the pink arrow).

Figure 4(d) shows the $f_d()$ function. Maximizing this function corresponds to choosing a $\overline{z}$ location which is farthest away from the current location.

Figure 4(e) shows the indicator function $f_h()$. In this scenario, the set $\tilde{B}_{max}$ is the set of cell centers which maximize $\hat{x}_w(k+d, \tilde{B})$. These correspond to the cell centers of the dark red rectangle in Figure 4(b). From the set $\tilde{B}_{max}$, $\overline{z}_H$ is chosen as the cell that is closest to the agent (the lower left corner point). Finally, the point $\overline{z}_h$ is determined as the point within the agent's reachable cell centers which is closest to $\overline{z}_H$. As can be seen, maximizing $f_h()$ requires simply choosing $\overline{z}$ in the same cell as $\overline{z}_h$.

Figure 4(f) shows the normalized total reward function. The reward function can typically exceed 1 but it is normalized so that the max value is 1 for plotting purposes.

The parameters α, β, γ, and δ can be chosen differently to reflect different agent capabilities. For example, small α values yield agents which will not be adverse to large course changes. This may be appropriate for more agile agents such as small UAVs. Conversely, large α values correspond to agents which may deem a coordinate more desirable if it has a somewhat low score but is in line with its current course. This may be

appropriate for cumbersome agents like large boats. One important aspect of Eq. 10 is that α, β, γ, δ, $\hat{x}_w()$, $f_\chi()$, $f_d()$, and $f_h()$ are all in the range $[0,1]$. This is explained further in Section III.C.2.

Previous work[23] investigated using adaptive sampling methods to provide quasi-optimal solutions of Eq. 10 over the feasible set B_R. Although this method worked well for most cases, it was nondeterministic. The stochastic nature when selecting $\overline{z}^\star$ makes analysis difficult. Therefore, instead of optimizing $J_0()$ over the compact set B_R, the problem is reduced to optimizing over the feasible set of $\tilde{B}_R$. With this approximation, the most desirable cell is then given by

$$(\wp_2) \quad \overline{z}^\star \in \arg\underset{\overline{z}\in\tilde{B}_R}{\text{maximize}}\ J_0(\overline{z}) \tag{21}$$

The problem formulated in Eq. 21 differs in that the feasible set is manageably finite. Therefore, the optimization is reduced to an exhaustive search over the set $\tilde{B}_R$. An additional benefit of this method is that it can be easily implemented by evaluating the reward function at each cell center in the set $\tilde{B}_R$.

C. Exhaustive Searching

The primary goal of a search mission is to find one or more targets which are located somewhere in the domain. One would be concerned if there are conditions where the target might be able to hide in the environment and avoid detection by the agents. This paper shows that under a reasonable set of assumptions, the agents are guaranteed to visit all cells in the map with non-zero score sufficiently often to drive the cell scores to zero. In other words, they will exhaustively search and cover the entire area of interest. The assumptions are

$$\begin{aligned}
\text{Assumptions:}\quad & h \in (0,1) && (A.1)\\
& \delta \in (0,1] && (A.2)\\
& R_{max} \geq \max(L_x, L_y) && (A.3)\\
& Z_k = z_A \ \ \forall k && (A.4)\\
& \hat{x}_w(k+d,\overline{z}) = x_w(k,\overline{z}) \ \ \forall k && (A.5)\\
& x_w(0,\overline{z}) \in [0,1) \ \ \forall \overline{z} && (A.6)\\
& \delta > \beta + \gamma && (A.7)
\end{aligned}$$

The physical meaning of assumption A.1 is to ensure that each agent has a sensor that can be relied upon to some degree. Eliminating the possibility that $h = 0$ ensures that with each sensor measurement, the score of the searched cell decreases. Note that the case of $h = 1$ implies an infinitely reliable sensor which is actually the best scenario. However, this requires rewording several following theorems so it is simply excluded using this set of assumptions. All of the following analysis can apply for the case of $h = 1$ with slight changes.

Recall from Section III.B, the scalars α, β, γ, and δ are in the range of $[0,1]$. Assumption A.2 ensures that $\delta \neq 0$. Its significance will become clear in Section III.C.2.

Assumption A.3 effectively specifies a minimum size of the set $\tilde{B}_R$. This states that there must be more than one cell center in $\tilde{B}_R$. Furthermore, this guarantees that $\tilde{B}_R$ includes the cells centers to the North, East, South, and West of the current agent location and ensures that the agent can move in any direction and eventually reach any other cell under an appropriate control law.

Assumptions A.4, A.5, and A.6 deal with the nature of the complete coverage of the map. Complete coverage implies the agents will search each cell in the map sufficiently to drive its score to zero. In a scenario involving complete coverage of the map, the target is located in the last cell that the agents would search or is simply not located in the map at all. These assumptions ensure that the scores of the map are never increasing and can in fact decrease under Eq. 4.

Assumption A.7 is crucial for the proof of complete coverage and is explained in Section III.C.2.

All of these assumptions are reasonable and can be implemented easily. Under these assumptions we will show that the agents operating under this strategy will exhaustively search the map and drive all cell scores towards zero given sufficient time.

1. Decreasing Scores

As the agents search a cell, it is desired that the score of the cells decrease. We show that the scores of each cell monotonically decrease with each sensor measurement.

Theorem III.1. *Under assumptions A.1, A.4, A.5, and A.6 and the previously described and search strategy, the score of any given occupancy map cell with a score not equal to* 0 *or* 1 *will monotonically decrease towards 0.*

Proof. Recall that the score of the cell is updated via Eq. 4 with $Z_k = z_A \ \forall k$. It is trivial to see that if $s_{k-1} = 1$ then $s_k = 1$ and if $s_{k-1} = 0$, then $s_k = 0$. The burden is to now show that $s_k < s_{k-1} \ \forall k$ for the cases where $s_{k-1} \in (0, 1)$. This was already proved in Theorem II.1

It is trivial to show that $s_k > 0 \ \forall k$. This, in conjunction with Eq. 9, shows that with enough sensor measurements of $Z_k = z_A$, the score of a given cell will proceed monotonically towards 0. □

2. Guaranteed Coverage

Theorem III.1 shows that if a cell is searched a sufficient number of times, its score will decrease towards zero. In order to show that the agents exhaustively search the map, the burden is to show that under the given control law, the agent will visit each cell enough times to decrease the scores to zero.

Theorem III.2. *Under assumptions A.3, A.4, A.5, and the previously described search strategy, if an agent is not currently in the cell containing $\overline{z}_H$, the agent must move to a new cell once the quantity $\alpha x_w(k, \overline{z}_{agt})$ decreases below the value δ.*

Proof. If the agent is not in the cell containing $\overline{z}_H$, then it cannot be at the location $\overline{z}_h$ either since A.3 $\Leftrightarrow$ $|\tilde{B}_R| > 1$ and by definition of $\overline{z}_h$, $\overline{z}_h$ must be closer to $\overline{z}_H$ than the current agent position. This implies that $f_h(\overline{z}_{agt}) = 0$. Similarly, by definition in Eq. 13, $f_\chi(\overline{z}_{agt}) = 0$. The maximum possible reward gained by choosing the same cell as the current agent (denoted $\overline{z}_s$) is

$$J_0(\overline{z}_s) = \alpha x_w(k, \overline{z}_s) + \eta\gamma f_d(\overline{z}_s) \tag{22}$$

By definition, $\overline{z}_h \in \tilde{B}_R$. It is possible that $x_w(k, \overline{z}_h) = f_\chi(\overline{z}_h) = 0$ (if the score is already zero and the cell is located directly behind the agent) so the minimum possible value of the reward function at this point is

$$J_0(\overline{z}_h) = \eta\gamma f_d(\overline{z}_h) + \delta \tag{23}$$

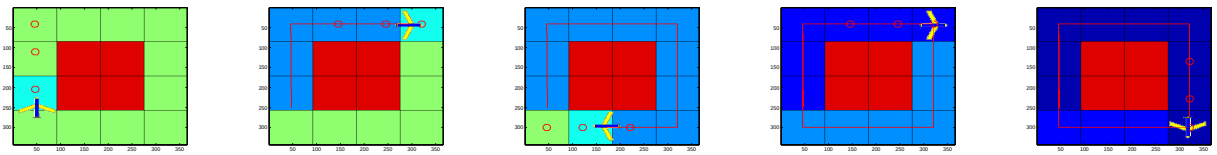
If the control solution from ($\wp_3$) performs correctly, the agent should be located at the point $\overline{z}^\star$ from ($\wp_2$) and $f_d(\overline{z}_s) = 0$. However, the proof can be applied for the case where the agent is not necessarily located on a cell center at all times. From a geometric perspective, the maximum value of $f_d(\overline{z}_s) = (L_x^2 + L_y^2)^{1/2}/R_{max}$ which corresponds to the agent located in the corner of the cell. The distance to any other cell in $\tilde{B}_R$ must be at least that if not greater, so $f_d(\overline{z}_h) \geq f_d(\overline{z}_s)$. This result, combined with Eq. 22 and Eq. 23, yields the result that

$$\{\alpha x_w(k, \overline{z}_s) < \delta\} \Rightarrow \{J_0(\overline{z}_h) > J_0(\overline{z}_s)\} \tag{24}$$

□

This shows the existence of a point with a higher score than the agent's current cell. This gives a lower bound of the score when the agent must choose to leave the current cell. It is possible that the agent will chose a different cell before $\alpha x_w(k, \overline{z}_s) < \delta$.

Theorems III.1 and III.2 ensure that an agent searching a non-zero score cell will decrease the score consistently and an agent cannot remain at a given cell indefinitely. Combined, these two ensure movement of the agent to different cells but does not guarantee that the map is covered. In fact, there are counter examples where an agent can constantly choose cells of nearly zero score and never search all of the map, thereby ignoring certain cells which have non-zero scores. So far, the previous theorems only required assumptions A.1 - A.6. The example in Figure 5 shows that although assumptions A.1- A.6 may be satisfied, this is not sufficient to guarantee exhaustive map coverage.



(a) $t = 0$ (b) $t = 195$ (c) $t = 325$ (d) $t = 520$ (e) $t = 845$

Figure 5. Example with $\alpha = 0.15$, $\beta = 0.95$, $\gamma = 0.85$, $\delta = 0.10$ showing a map not being covered due to assumption A.7 violation.

In this example, the agent continues to fly around the perimeter of the map and never explores the interior. This is due to the fact that the cost of changing heading to turn into the interior of the map is too high relative to the other terms in the cost function. In order to guarantee complete coverage of the map, the restriction of A.7 must be enforced.

Theorem III.3. *Under assumption A.7 and the previously described search strategy, the agent must choose either a cell which has non-zero score, contains $\overline{z}_h$, or both as the solution to $(\wp_2)$.*

Proof. Assume there exists a point $\overline{z}_0$ which has a score of zero and is not $\overline{z}_h$. In this case $x_w(k, \overline{z_0}) = f_h(z_0) = 0$. The maximum value of the reward function at this point is

$$J_0(\overline{z}_0) = \eta(\beta + \gamma) \tag{25}$$

The minimum value of the reward function obtained by choosing $\overline{z}_h$ is

$$J_0(\overline{z}_h) = \delta \tag{26}$$

Since $\eta \in [0, 1]$, these two results along with the assumption $\delta > \gamma + \beta$ ensures that $J_0(\overline{z}_h) > J_0(\overline{z}_0)$. This guarantees that the agent will not choose a cell if it has zero score and is not $\overline{z}_h$. Therefore, the only other options are to choose a cell which has a non-zero score, contains $\overline{z}_h$, or both. □

Theorem III.4. *Under assumptions A.4, A.5 and the previously described search strategy, if Eq. 16 is used to choose $\overline{z}_H$ and the agent chooses $\overline{z}_h$ as the solution to $(\wp_2)$ and $\overline{z}_h \neq \overline{z}_H$, then at the next time step the point $\overline{z}_H$ will remain unchanged unless it is searched by another agent.*

Proof. Consider the distance between $\overline{z}_H$ and $\overline{z}_{agt}$ at a given time step k ($||\overline{z}_H - \overline{z}_{agt}||$). By definition, $\overline{z}_h \neq \overline{z}_H \Leftrightarrow \overline{z}_H \notin \tilde{B}_R$. So if the agent chooses $\overline{z}_h$ as the solution to $(\wp_2)$ and $\overline{z}_h \neq \overline{z}_H$, the agent must move closer to the point $\overline{z}_H$ in the sense that $||\overline{z}_H - \overline{z}_{agt}||$ cannot increase. Since the point $\overline{z}_h$ is not equal to $\overline{z}_H$ then the agent cannot search the cell containing $\overline{z}_H$ and score at $\overline{z}_H$ will not change. Therefore, at the next time step, the same point $\overline{z}_H \in \tilde{B}_{max}$. Furthermore, since the distance between the agent and the same point $\overline{z}_H$ has decreased, it will be chosen again as $\overline{z}_H$ using Eq. 16.

Of course, if the point $\overline{z}_H$ is searched first by another agent, its score may decrease and it is possible that at the next step the same point is no longer in $\tilde{B}_{max}$ due to the fact that the score decreased. □

A similar theorem can be stated for the case where $\overline{z}_H$ is chosen using Eq. 17.

Theorem III.5. *Under assumptions A.4, A.5 and the previously described search strategy, if Eq. 17 is used to choose $\overline{z}_H$ and the agent chooses $\overline{z}_h$ as the solution to $(\wp_2)$ and $\overline{z}_h \neq \overline{z}_H$, then at the next time step the point $\overline{z}_H$ will either remain unchanged or be located in the agent's reachable set, $\tilde{B}_R$ unless it is searched by another agent.*

Proof. Recall that if $\tilde{B}_{max} \bigcap \tilde{B}_R = \emptyset$ (there are no cells with maximum score within the agent's reachable set), then both Eq. 16 and Eq. 17 choose the same cell center for $\overline{z}_H$, so the proof of Theorem III.4 applies and shows that as long as $\tilde{B}_{max} \bigcap \tilde{B}_R = \emptyset$, the point $\overline{z}_H$ will remain unchanged under these assumptions. The difference between Eq. 16 and Eq. 17 occurs when $\tilde{B}_{max} \bigcap \tilde{B}_R \neq \emptyset$.

Assume at step k, the $\tilde{B}_{max} \bigcap \tilde{B}_R = \emptyset \Rightarrow \overline{z}_H \notin \tilde{B}_R$. Assuming that the agent chooses $\overline{z}_h$ as the solution to $(\wp_2)$ and $\overline{z}_h \neq \overline{z}_H$, the previous analysis shows that $\overline{z}_H$ does not change. However, at the next step $k+1$, if the same point $\overline{z}_H \in \tilde{B}_R$, then it is possible that Eq. 17 will not choose the same point $\overline{z}_H$ again because

there might be another member of $\tilde{B}_{max} \bigcap \tilde{B}_R$ which is farther away from $\overline{z}_{agt}$. However, it is guaranteed that $\tilde{B}_{max} \bigcap \tilde{B}_R \neq \emptyset$ since the same point $\overline{z}_H$ from step k is in $\tilde{B}_{max}$ and $\tilde{B}_R$ under these assumptions, thus ensuring that $\overline{z}_H \in \tilde{B}_R$.

Of course, if the point $\overline{z}_H$ is searched first by another agent, its score may decrease and it is possible that at the next step the same point is no longer in $\tilde{B}_{max}$ due to the fact that the score decreased. □

Figure 3 can be used as a visual aid for understanding the proofs of both Theorems III.4 and III.5. These theorems can be used to guarantee that the team performs an exhaustive search of the map given the proper conditions.

Theorem III.6. *Under the previously described assumptions and search strategy,* $x_w(k, \overline{z}) \to 0 \ \forall \overline{z} \in B$ *(the scores of all cells in the map will approach 0).*

Proof. Theorem III.2 guarantees that an agent cannot remain in a single cell indefinitely and Theorem III.3 ensures that it must choose either a cell of non-zero score, $\overline{z}_h$, or both as the solution to $(\wp_2)$ at any given time step. If the agent chooses a cell of non-zero score, Theorem III.1 ensures that the score of that cell is monotonically decreased towards 0.

If the agent does not choose a cell of non-zero score, the only alternative scenario allowed by Theorem III.3 is that the agent chooses $\overline{z}_h$ and $x_w(k, \overline{z}_h) = 0$. By definition of $\overline{z}_H$, if $x_w(k, \overline{z}_H) = 0$, then the map has been completely covered since all scores are at most 0. Therefore, assuming that the map has not been entirely searched yet, $x_w(k, \overline{z}_h) = 0 \Rightarrow \overline{z}_h \neq \overline{z}_H$.

Theorem III.4 and Theorem III.5 ensure that if $\overline{z}_h \neq \overline{z}_H$, choosing $\overline{z}_h$ as the solution to $(\wp_2)$ does not change the value of $\overline{z}_H$ at the next time step or if $\overline{z}_H$ does change, $\overline{z}_H \in \tilde{B}_R$ at the next time step.

At this next time step, the agent once again must choose a cell with non-zero score, $\overline{z}_h$, or both. If the agent continues to choose $\overline{z}_h$, eventually $\overline{z}_H \in \tilde{B}_R$ and at this point, choosing a cell with non-zero score, $\overline{z}_h$, or both guarantees that a cell of non-zero score is chosen. Therefore, the score of some cell will be ensured to decrease and given sufficient time, $x_w(k, \overline{z}) \to 0 \ \forall \overline{z} \in B$.

□

Note that the results of Theorem III.6 yield a stronger result than simply ensuring that each cell is visited during the mission. In the specified framework, visiting a cell once may not be enough to guarantee that the target is not located in that cell. In the case where the agent has an unreliable sensor, the amount of information obtained from a single visit to a cell may not decrease the score of that cell sufficiently. Theorem III.6 and its associated proof show that the map will be exhaustively searched for the target using any number of agents in the sense that each cell is visited a sufficient number of times to drive the scores of all cells in the map to zero.

Note that only one agent must be constricted by assumption A.7 in order to guarantee an exhaustive search by the team. Performance may be increased by relaxing assumption A.7 for some agents but it cannot be guaranteed that the map will be covered by the agents who do not satisfy assumption A.7

IV. Modifications for Explicit Cooperation

The search strategy described in Section III was derived as a single agent search strategy, each agent in the team follows this same policy without information or explicit knowledge of other members of the team. This section investigates some modifications to the single agent search strategy to allow for explicit cooperation between agents in the team. This has the potential to increase performance and change the behavior of the system although it requires additional complexity and computational resources and may diminish the scalability of the algorithm and possibly make it impractical for large numbers of agents.

A. $(\wp_2)$ with Explicit Cooperation

Recall that solving $(\wp_2)$ involved picking a desirable location for the agent to search within the agent's reachable set, B_R. Without explicit cooperation, there are scenarios where two or more agents might choose the same location $\overline{z}^\star$. This is undesirable for several reasons. One is that the agents must now have some type of reactive collision avoidance[34,35] since collision free trajectories are not guaranteed at the planning level. Furthermore, the performance of the system may suffer if there are multiple agents searching the same cells. One method to alleviate these problems is to partition the search space. In essence, this is a divide

and conquer approach. The search space can be partitioned using a Voronoi diagram which is a tessellation of a Euclidean space and is described in a purely mathematical sense by Okabe[36] . This section investigates applying them to ($\wp_2$). In this sense, the location of the agents are considered to be generators for the Voronoi polygons

$$P = \{p_1, p_2, ..., p_n\} = \{\overline{z}_{agt_1}, \overline{z}_{agt_2}, ..., \overline{z}_{agt_n}, \} \tag{27}$$

The Voronoi diagram embeds information about an agent's position relative to the other agents. Each agent now has an influence on the diagram and each agent will have an influence on the algorithm. The degradation of algorithm scalability is immediately visible when employing this type of cooperation as the Voronoi diagram requires at least $O(n\ log\ n)$ at best[36] where n is the number of generators (or agents). Previously, the single agent search strategy scaled linearly with n.

1. *Redefining $\overline{z}_{h_i}$ with Voronoi Diagrams*

The point in $\tilde{B}_{max}$ that is closest to agent i is given as

$$\overline{z}_{H_i} \in \underset{\overline{z} \in \tilde{B}_{max}}{\text{arg minimize}} ||\overline{z} - \overline{z}_{agt_i}|| \tag{28}$$

With $\overline{z}_{H_i}$ defined, the distance between agent i and $\overline{z}_{H_i}$ is given by

$$d_i = ||\overline{z}_{H_i} - \overline{z}_{agt_i}|| \tag{29}$$

An example of these distances and locations is shown in Figure 6. In this figure, the black dots represent $\tilde{B}$ (the cell centers of the occupancy map). The black dots circled in purple represent $\overline{z} \in \tilde{B}_{max}$. These are the cell centers that correspond to cells which have the highest score in the map. As defined previously, locations inside the dashed red circles represent B_{R_i} (agent i's reachable set). The black dots within the dashed red circles represent $\tilde{B}_{R_i}$ (the discrete approximation of agent i's reachable set).

The distance d_i is computed by checking the distance between agent i and each $\overline{z} \in \tilde{B}$ with the minimum value assigned as d_i. The corresponding location $\overline{z} \in \tilde{B}$ which yields d_i for agent i is denoted as $\overline{z}_{H_i}$.

Note that $\overline{z}_{H_i}$ may not be in the agent's reachable set (either the compact set B_{R_i} or its discrete approximation $\tilde{B}_{R_i}$). In other words, it is possible that $\overline{z}_{H_i} \notin B_{R_i}$ and $\overline{z}_{H_i} \notin \tilde{B}_{R_i}$. This is the case with both agent 1 and agent 2 but not agent 3 in Figure 6.

The Voronoi edges are shown as the solid black lines. The Voronoi polygon associated with each agent is enclosed by the solid black lines. Note that it is possible that $\overline{z}_{H_i} \notin V(\overline{z}_{agt_i})$. In Figure 6, this is the case for agent 3 where the point $\overline{z}_{H_3}$ is not in agent 3's Voronoi polygon. It is also possible that $\overline{z}_{H_i} = \overline{z}_{H_j}$ for some $i \neq j$ as shown with agent 2 and agent 3 in Figure 6.

In general, $a \notin A$ does not imply that $a \notin B$ if $A \subset B$. However, in the case of $\overline{z}_{H_i}$ it can be shown that $\overline{z}_{H_i} \notin \tilde{B}_R \Rightarrow \overline{z}_{H_i} \notin B_R$

Theorem IV.1. *If $\overline{z}_{H_i}$ is computed using Eq. 28, $\overline{z}_{H_i} \notin \tilde{B}_R \Rightarrow \overline{z}_{H_i} \notin B_R$.*

Proof. Using the definitions of B and $\tilde{B}$ from Section II.A and the definitions of B_{max}, B_{R_i}, $\tilde{B}_{R_i}$ from Section **??**, the relationships of the sets can be drawn as shown in Figure 7.

From Eq. 28 it can be seen that $\overline{z}_{H_i} \in \tilde{B}_{max}$. $\tilde{B}_{max}$ is denoted by vertical lines in Figure 7. The set $\tilde{B}_{R_i}$ is shown as the grey shaded area in Figure 7. Therefore, $\overline{z}_{H_i} \in \tilde{B}_{max}$ and $\overline{z}_{H_i} \notin \tilde{B}_R$ is only satisfied if $\overline{z}_{H_i}$ is in the subset of $\tilde{B}_{max}$ which is marked with the vertical lines with no grey shading. The intersection of this set and B_{R_i} is empty since the sets $\tilde{B}_{max}$ and $\tilde{B}_{R_i}$ are closed. Therefore, $\overline{z}_{H_i} \notin \tilde{B}_R \Rightarrow \overline{z}_{H_i} \notin B_R$ □

Most of the following analysis requires identifying the agent which is closest to the set $\tilde{B}_{max}$. The index of the agent closest

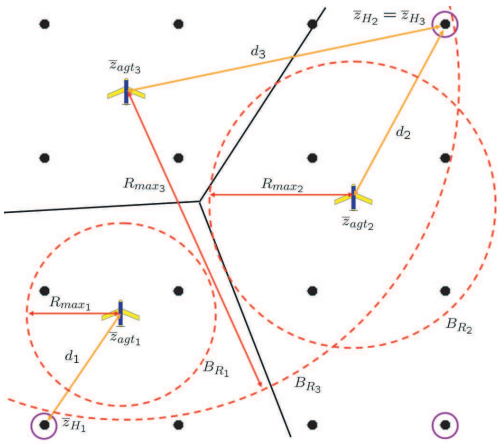


Figure 6. Example with 3 agents showing d_i and $\overline{z}_{H_i}$.

to the set $\tilde{B}_{max}$ is denoted as I. The index I is given by (recall that $I_n = \{1, 2, ..., n\}$)

$$I \in \underset{i \in I_n}{\text{arg minimize}}\ d_i \tag{30}$$

Therefore, $\overline{z}_{H_I}$ denotes the location of the point in $\tilde{B}_{max}$ that is closest to any agent.

As stated previously, in general it is possible that $\overline{z}_{H_i} \notin V(\overline{z}_{agt_i})$. However it is true that $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$ always.

Theorem IV.2. *Given any Voronoi diagram* $\overline{V} = \{V(\overline{z}_{agt_1}), ..., V(\overline{z}_{agt_n})\}$, $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$.

Proof. As stated previously, $\overline{z}_{H_I}$ denotes the location of the point in $\tilde{B}_{max}$ which is closest to any agent. Denote this agent's index as I and its location as $\overline{z}_{agt_I}$.

Note that the definition of the Voronoi polygon generated by agent I is given as

$$V(\overline{z}_{agt_I}) = \{\overline{z} : ||\overline{z} - \overline{z}_{agt_I}|| \leq ||\overline{z} - \overline{z}_{agt_i}|| \ \text{ for } i \in I_n, i \neq I\} \tag{31}$$

In order for $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$, the following inequality must be satisfied

$$||\overline{z}_{H_I} - \overline{z}_{agt_I}|| \leq ||\overline{z}_{H_I} - \overline{z}_{agt_i}|| \ \text{ for } i \in I_n, i \neq I \tag{32}$$

By definition, $d_I \leq d_i$ for $i \in I_n$. Therefore, $\overline{z}_{H_I}$ has the property that

$$||\overline{z}_{H_I} - \overline{z}_{agt_I}|| \leq ||\overline{z}_{H_i} - \overline{z}_{agt_i}|| \ \text{ for } i \in I_n \tag{33}$$

From Eq. 28, it can be seen that $\overline{z}_{H_i}$ is the point in $\tilde{B}_{max}$ which is closer to agent i than any other point in $\tilde{B}_{max}$. In other words, $\overline{z}_{H_i}$ has the property that $||\overline{z}_{H_i} - \overline{z}_{agt_i}|| \leq ||\overline{z} - \overline{z}_{agt_i}||$ for all $\overline{z} \in \tilde{B}_{max}$. Choosing $\overline{z}_{H_I} \in \tilde{B}_{max}$ in the previous expression yields the relationship

$$||\overline{z}_{H_i} - \overline{z}_{agt_i}|| \leq ||\overline{z}_{H_I} - \overline{z}_{agt_i}|| \text{ for } i \in I_n \tag{34}$$

Overbounding Eq. 33 with Eq. 34 yields

$$||\overline{z}_{H_I} - \overline{z}_{agt_I}|| \leq ||\overline{z}_{H_I} - \overline{z}_{agt_i}|| \ \text{ for } i \in I_n \tag{35}$$

The inequality given in Eq. 32 only needs to hold for $i \in I_n \setminus \{I\}$ whereas the result in Eq. 35 is valid for $i \in I_n$. Therefore, this shows that $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$. □

Remark IV.3. *Theorem IV.2 can be argued intuitively as follows. The quantity* $d_i = ||\overline{z}_{H_i} - \overline{z}_{agt_i}||$ *is the shortest distance between* $\overline{z}_{agt_i}$ *and the set* $\tilde{B}_{max}$. *Without loss of generality, consider there to be only a single point in the map which has maximum score. In other words,* $\tilde{B}_{max} = \overline{z}_H = \overline{z}_{H_i} \forall i$. *In this case,* d_i *is the distance between this point and generator* $\overline{z}_{agt_i}$. *In general,* $\overline{z}_{H_i}$ *is not necessarily in* $V(\overline{z}_{agt_i})$ *(i.e.* $\overline{z}_{H_3}$ *in Figure 6). For the case of* $i = I$, *the generator* $\overline{z}_{agt_I}$ *is closer to* $\overline{z}_H$ *than any other generator by definition. Therefore, by definition of the Voronoi polygon* $V(\overline{z}_{agt_I})$, *the point* $\overline{z}_H = \overline{z}_{H_I} \in V(\overline{z}_{agt_I})$.

Recall that the function $f_h()$ (Eq. 19) served to indicate the direction the agent could travel in the set $\tilde{B}_R$ in order to move towards a cell of maximum score. This required definition of the point $\overline{z}_{h_i}$. The presence of the Voronoi polygon complicates matters slightly. In this context, the definition of $\overline{z}_{h_i}$ is through several intermediate variables. The first of which is

$$\overline{z}'_{h_i} \in \begin{cases} \underset{\overline{z} \in \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i})}{\arg \text{ minimize }} ||\overline{z} - \overline{z}_{H_i}|| & \text{if } \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) \neq \emptyset \\ \overline{z}_{agt_i} & \text{otherwise} \end{cases} \tag{36}$$

Although it is guaranteed that $V(\overline{z}_{agt_i})$ has finite area, it is not guaranteed that $\tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) \neq \emptyset$. Due to the proximity

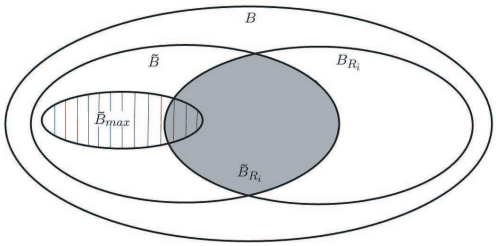


Figure 7. Relationships between sets B, $\tilde{B}$, $\tilde{B}_{max}$, B_{R_i}, and $\tilde{B}_{R_i}$.

of other agents, the Voronoi polygon $V(\overline{z}_{agt_i})$ may be small
and not include any of the points which make up $\tilde{B}_{R_i}$. Despite this fact, using Eq. 36, it is guaranteed that $\overline{z}'_{h_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$.

Theorem IV.4. *Given any Voronoi diagram* $\overline{V} = \{V(\overline{z}_{agt_1}), ..., V(\overline{z}_{agt_n})\}$, $\overline{z}'_{h_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$ *for all* $i \in I_n$.

Proof. If the set $\tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) \neq \emptyset$ then the feasible set of the minimization problem in Eq. 36 is $\tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i})$ and therefore, $\overline{z}'_{h_i} \in \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i})$. Since $\tilde{B}_R \subset B_{R_i}$, $\overline{z}'_{h_i} \in \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) \Rightarrow \overline{z}'_{h_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$.

If the set $\tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) = \emptyset$ then $\overline{z}'_{h_i} = \overline{z}_{agt_i}$. By definition of a Voronoi polygon, $\overline{z}'_{h_i} = \overline{z}_{agt_i} \in V(\overline{z}_{agt_i})$ since $\overline{z}_{agt_i}$ is the generator for the Voronoi polygon $V(\overline{z}_{agt_i})$. By definition of B_{R_i} (Eq. 11), $\overline{z}'_{h_i} = \overline{z}_{agt_i} \in B_{R_i}$ for all possible values of R_{max} (since negative R_{max} values are not allowed in this framework). Since $\overline{z}'_{h_i} \in V(\overline{z}_{agt_i})$ and $\overline{z}'_{h_i} \in B_{R_i}$, then $\overline{z}'_{h_i} = \overline{z}_{agt_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$. □

Remark IV.5. *It should be noted that the reachable set in Theorem IV.4 is* B_{R_i}, *not* $\tilde{B}_{R_i}$.

With $\overline{z}'_{h_i}$ defined, it is convenient to define a flag ζ_i as

$$\zeta_i = \begin{cases} 1 & \text{if } ||\overline{z}'_{h_i} - \overline{z}_{H_i}|| \geq ||\overline{z}_{agt_i} - \overline{z}_{H_i}|| \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

Physically, $\zeta_i = 1$ if the point $\overline{z}'_{h_i}$ is further away from $\overline{z}_{H_i}$ than the agent's current position of $\overline{z}_{agt_i}$. An example showing when $\zeta_i = 1$ is shown in Figure 8. In this situation, the set $\tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i})$ is a single point and therefore, Eq. 36 chooses it as $\overline{z}'_{h_i}$ (denoted by the black dot enclosed by the orange circle). In this case, $\overline{z}'_{h_i}$ is obviously farther away from $\overline{z}_{H_i}$ than $\overline{z}_{agt_i}$, so $\zeta_i = 1$.

With the flag ζ_i defined, the point $\overline{z}_{h_i}$ is given by

$$\overline{z}_{h_i} = \begin{cases} \overline{z}'_{h_i} & \text{if } \zeta_i = 0 \\ \overline{z}_{agt_i} & \text{if } \zeta_i = 1 \text{ and } i \neq I \\ \overline{z}_{S_I} & \text{if } \zeta_i = 1 \text{ and } i = I \end{cases} \tag{38}$$

where

$$\overline{z}_{S_I} = \overline{z}_{agt_I} + R_{max_I} \frac{\overline{z}_{H_I} - \overline{z}_{agt_I}}{||\overline{z}_{H_I} - \overline{z}_{agt_I}||} \tag{39}$$

The point $\overline{z}_{h_i}$ has several interesting properties

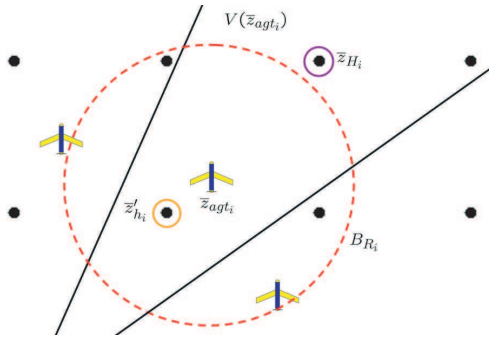


Figure 8. Situation showing $\zeta_i = 1$.

Theorem IV.6. *The point* $\overline{z}_{h_i}$ *is always reachable by agent* i *and remains in the agent's Voronoi polygon,* $\overline{z}_{h_i} \in B_R \bigcap V(\overline{z}_{agt_i})$.

Proof. Eq. 38 states that $\overline{z}_{h_i}$ can be equal to either $\overline{z}'_{h_i}$, $\overline{z}_{agt_i}$, or $\overline{z}_{S_I}$, depending on the situation. Showing $\overline{z}_{h_i} \in B_R \bigcap V(\overline{z}_{agt_i})$ can be achieved by showing that each of the three previously mentioned values are in this set as well (depending on the situation).

Theorem IV.4 showed that $\overline{z}'_{h_i} \in B_R \bigcap V(\overline{z}_{agt_i})$. The proof section of Theorem IV.4 also showed how $\overline{z}_{agt_i} \in B_R \bigcap V(\overline{z}_{agt_i})$.

The last case to check is if $\overline{z}_{h_i} = \overline{z}_{S_I}$. From Eq. 38, $\overline{z}_{h_i} = \overline{z}_{S_I}$ if and only if $i = I$ and $\zeta_I = 1$. For the case where $i = I$ and $\zeta_I = 1$, from Eq. 37, $\zeta_I = 1 \Rightarrow ||\overline{z}'_{h_I} - \overline{z}_{H_I}|| \geq ||\overline{z}_{agt_I} - \overline{z}_{H_I}||$. Looking at the equation for $\overline{z}'_{h_I}$ (Eq. 36), it can be seen that $\overline{z}'_{h_I} \neq \overline{z}_{agt_I}$ (or else ζ_I would equal 0 and this cannot happen in this case). Therefore, $\overline{z}'_{h_I}$ must be obtained by solving the minimization problem in Eq. 36. Furthermore, in order for $\zeta_I = 1$, the solution to the minimization problem must be farther away from the point $\overline{z}_{H_I}$ than the current agent location so $\overline{z}_{H_I}$ cannot be in the feasible set of $\tilde{B}_{R_I} \bigcap V(\overline{z}_{agt_I})$ (otherwise it would be chosen as the minimizer since it would yield a cost of 0 in the cost function). Theorem IV.2 showed that $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$ so $\overline{z}_{H_I} \notin \tilde{B}_{R_I}$. Theorem IV.1 can then be applied to show that $\overline{z}_{H_I} \notin \tilde{B}_{R_I} \Rightarrow \overline{z}_{H_I} \notin B_{R_I}$. Now note that $\overline{z}_{H_I} \notin B_{R_I} \iff ||\overline{z}_{H_I} - \overline{z}_{agt_I}|| > R_{max_I}$. Since $R_{max_I} > 0$, Eq. 39 can be written as

$$\overline{z}_{S_I} = \overline{z}_{agt_I} + \theta(\overline{z}_{H_I} - \overline{z}_{agt_I}) \in V(\overline{z}_{agt_I}) \text{ for some } \theta \in (0, 1) \text{ when } \zeta_I = 1 \tag{40}$$

Theorem IV.2 gave one point that is always in $V(\overline{z}_{agt_I})$ by showing that $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$. The generator of the Voronoi polygon is a second point which is always in the Voronoi polygon, so $\overline{z}_{agt_I} \in V(\overline{z}_{agt_I})$. Using the fact that all Voronoi polygons are convex sets[36] and that the line segment between any two points in a convex set is in the convex set as well, it can be seen that the line segment $\overline{z}_{agt_I} + \theta(\overline{z}_{H_I} - \overline{z}_{agt_I}) \in V(\overline{z}_{agt_I})$ for some $\theta \in [0, 1]$. Comparing this to the result in Eq. 40, it can be seen that

$$\overline{z}_{S_I} \in V(\overline{z}_{agt_I}) \text{ when } \zeta_I = 1 \tag{41}$$

Physically, the point $\overline{z}_{S_I}$ is obtained by moving a distance R_{max_I} towards $\overline{z}_{H_I}$ from the point $\overline{z}_{agt_I}$. So by construction, $\overline{z}_{S_I} \in B_{R_I}$. This, combined with the result in Eq. 41, shows that $\overline{z}_{S_I} \in B_{R_I} \bigcap V(\overline{z}_{agt_I})$ when $\zeta_I = 1$.

The combined analysis shows that $\overline{z}_{h_i} \in \overline{z}_{h_i} \bigcap V(\overline{z}_{agt_i})$ for all $i \in I_n$. □

The point $\overline{z}_{h_i}$ has the property that it is guaranteed to no be farther away from the point $\overline{z}_{H_i}$ than the agent's current location.

Theorem IV.7. *The point $\overline{z}_{h_i}$ is not farther away from the point $\overline{z}_{H_i}$ than the agent's current location in the sense that $||\overline{z}_{h_i} - \overline{z}_{H_i}|| \leq ||\overline{z}_{agt_i} - \overline{z}_{H_i}|| \; \forall i \in I_n \setminus \{I\}$ and $||\overline{z}_{h_I} - \overline{z}_{H_I}|| < ||\overline{z}_{agt_I} - \overline{z}_{H_I}||$ (the inequality is strict for the case of $i = I$).*

Proof. From Eq. 38 if $\zeta_i = 0$, then $\overline{z}_{h_i} = \overline{z}'_{h_i}$. From Eq. 37, $\zeta_i = 0 \iff ||\overline{z}'_{h_i} - \overline{z}_{H_i}|| < ||\overline{z}_{agt_i} - \overline{z}_{H_i}||$ so

$$||\overline{z}_{h_i} - \overline{z}_{H_i}|| < ||\overline{z}_{agt_i} - \overline{z}_{H_i}|| \text{ if } \zeta_i = 0 \text{ and } \overline{z}_{h_i} = \overline{z}'_{h_i} \tag{42}$$

For the case when $\zeta_i = 1$ and $i \neq I$, then $\overline{z}_{h_i} = \overline{z}_{agt_i}$. In this situation it is obvious that

$$||\overline{z}_{h_i} - \overline{z}_{H_i}|| = ||\overline{z}_{agt_i} - \overline{z}_{H_i}|| \text{ if } \zeta_i = 1 \text{ and } \overline{z}_{h_i} = \overline{z}_{agt_i} \tag{43}$$

Finally, for the case when $\zeta_i = 1$ and $i = I$, then $\overline{z}_{h_I} = \overline{z}_{S_I}$. The proof section of Theorem IV.6 showed that in this situation, $\overline{z}_{H_I} \notin B_{R_I}$ and Eq. 40 showed that $\overline{z}_{S_I}$ must be closer to $\overline{z}_{H_I}$ than $\overline{z}_{agt_I}$. Therefore,

$$||\overline{z}_{h_I} - \overline{z}_{H_I}|| < ||\overline{z}_{agt_I} - \overline{z}_{H_I}|| \text{ if } \zeta_I = 1 \text{ and } \overline{z}_{h_I} = \overline{z}_{S_I} \tag{44}$$

□

Remark IV.8. *Theorem IV.7 can also be written as $||\overline{z}_{h_i} - \overline{z}_{H_I}|| = ||\overline{z}_{agt_i} - \overline{z}_{H_I}||$ if $\zeta_i = 0$ and $i \neq I$. Otherwise, $||\overline{z}_{h_i} - \overline{z}_{H_I}|| < ||\overline{z}_{agt_i} - \overline{z}_{H_I}||$.*

The above analysis shows that the point $\overline{z}_{h_i}$ has some interesting properties:

1. $\overline{z}_{h_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$
2. $||\overline{z}_{h_i} - \overline{z}_{H_i}|| \leq ||\overline{z}_{agt_i} - \overline{z}_{H_i}||$ for $i \in I_n \setminus \{I\}$
3. $||\overline{z}_{h_I} - \overline{z}_{H_I}|| < ||\overline{z}_{agt_I} - \overline{z}_{H_I}||$ for $i = I$

These properties can be used to develop the control law with explicit cooperation between agents.

2. *Control Law with Explicit Cooperation*

The control law for choosing $\overline{z}^\star$ was previously described in Section III.C, Eq. 21. This involved maximizing the reward function $J_{0_i}()$ over the set $\tilde{B}_{R_i}$. The added complexity of the Voronoi diagram requires a slight redefinition of the reward function

$$\hat{J}_{0_i} = \alpha \hat{x}_w(k + d, \overline{z}) + \eta \left(\beta f_\chi(\overline{z}) + \gamma f_d(\overline{z})\right) \tag{45}$$

In Eq. 45 it is understood that all of the parameters α, β, and γ and all the functions η, $f_\chi()$, and $f_d()$ are specific to agent i. The variable η is defined slightly differently as

$$\eta = \max_{\overline{z} \in \tilde{B}_R \bigcap V(\overline{z}_{agt_i})} \hat{x}_w(k + d, \overline{z}) \tag{46}$$

Notice that this reward function omits the term that contains the indicator function $f_h()$. Previously, this was used to guarantee coverage of the map and it is instead incorporated in the algorithm in a heuristic fashion.

In order to maintain the Voronoi partitioning, the modified control law to incorporate explicit cooperation is now of the form

$$\overline{z}_i^{\star'} \in \begin{cases} \arg \underset{\overline{z} \in \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i})}{\text{maximize}} \hat{J}_{0_i}(\overline{z}) & \text{if } \tilde{B}_{R_i} \bigcap V(\overline{z}_{agt_i}) \neq \emptyset \\ \overline{z}_{agt_i} & \text{otherwise} \end{cases} \tag{47}$$

$$\overline{z}_i^{\star} \in \begin{cases} \overline{z}_{h_i} & \text{if } \alpha \hat{x}_w(k+d, \overline{z}_i^{\star'}) < \delta \\ \overline{z}_i^{\star'} & \text{otherwise} \end{cases} \tag{48}$$

Agent i chooses $\overline{z}_i^{\star}$ as the next location to search and the process repeats once the agent reaches $\overline{z}_i^{\star}$.

Theorem IV.9. *The point $\overline{z}_i^{\star}$ is always reachable by agent i and remains in the agent's Voronoi polygon, $\overline{z}_i^{\star} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$.*

Proof. In the case where $\overline{z}_i^{\star} = \overline{z}_i^{\star'}$, it is obvious From Eq. 47 that $\overline{z}_i^{\star'} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$ since the feasible set of the maximization problem is precisely $B_{R_i} \bigcap V(\overline{z}_{agt_i})$ and it was previously shown that $\overline{z}_{agt_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$.

In the case where $\overline{z}_i^{\star} = \overline{z}_{h_i}$, Theorem IV.6 can be applied to show that $\overline{z}_{h_i} \in B_{R_i} \bigcap V(\overline{z}_{agt_i})$. □

If the same assumptions in Section III.C are satisfied, several guarantees about algorithm performance can be made. However, unlike the results in Section III.C where guarantees are shown for all agents in the team, most of the guarantees for the case of explicit cooperation can be shown only for the case of $i = I$.

Theorem IV.10. *Under assumptions A.3, A.4, A.5, and the previously described search strategy with explicit cooperation, if agent I is not currently at the point $\overline{z}_{H_I}$ the agent must move to a point other than $\overline{z}_{agt_I}$ once the quantity $\alpha x_w(k, \overline{z}_{agt_I})$ decreases below the value δ.*

Proof. Assuming that $\overline{z}_i^{\star'} = \overline{z}_{agt_I}$, if the quantity $\alpha x_w(k, \overline{z}_{agt_I})$ decreases below the value δ, then Eq. 48 will assign $\overline{z}_i^{\star} = \overline{z}_{h_i}$. Theorem IV.7 can be applied to show that $\overline{z}_i^{\star}$ is strictly closer to $\overline{z}_{H_i}$ than $\overline{z}_{agt_I}$. Since the agent is not at the point $\overline{z}_{H_I}$, the agent must move to a point other than $\overline{z}_{agt_I}$. □

Theorem IV.10 shows that the agent cannot remain at the same location indefinitely. The control law in Eq. 48 differs from that in Section III.C. One consequence of this is that assumption A.7 is not required to ensure exhaustive searches. This is illustrated in the next several theorems.

Theorem IV.11. *Under the previously described search strategy, agent I must choose either a cell center which has non-zero score, $\overline{z}_{h_I}$, or both as the point $\overline{z}_I^{\star}$.*

Proof. The point $\overline{z}_i^{\star}$ is assigned according to Eq. 48. If the point $\overline{z}_i^{\star'}$ has a score of zero, then $\alpha \hat{x}_w(k+d, \overline{z}_i^{\star'}) = 0$ which is less than the value of δ for any $\delta \in (0, 1]$ so Eq. 48 will assign $\overline{z}_i^{\star} = \overline{z}_{h_i}$.

Alternatively, if $\alpha \hat{x}_w(k+d, \overline{z}_i^{\star'}) \geq \delta$, then Eq. 48 will assign $\overline{z}_i^{\star} = \overline{z}_i^{\star'}$. Since $\delta, \alpha \in (0, 1]$, $\alpha \hat{x}_w(k+d, \overline{z}_i^{\star'}) \geq \delta \iff \hat{x}_w(k+d, \overline{z}_i^{\star'}) > 0$ which shows that the point associated with $\overline{z}_I^{\star}$ has a non-zero score. □

Theorem IV.12. *Under assumption A.4, A.5 and the previously described search strategy, if agent I chooses $\overline{z}_{h_I}$ as $\overline{z}_I^{\star}$ and $\overline{z}_{h_I} \neq \overline{z}_{H_I}$, then at the next time step the point $\overline{z}_{H_I}$ will remain unchanged and some agent $i \in I_n$ will move closer to the point $\overline{z}_{H_I}$.*

Proof. Theorem IV.7 showed that $||\overline{z}_{h_I} - \overline{z}_{H_I}|| < ||\overline{z}_{agt_I} - \overline{z}_{H_I}||$ so if the agent chooses $\overline{z}_I^{\star} = \overline{z}_{h_I}$, at the next time step, it be closer to the point $\overline{z}_{H_I}$ than it was before. If $\overline{z}_{h_I} \neq \overline{z}_{H_I}$, then the agent cannot search that cell so the score will not decrease and at the next step, $\overline{z}_{H_I}$ will still be in the set $\tilde{B}_{max}$. Unlike Theorem III.4, another agent cannot search this same location due to fact that $\overline{z}_i^{\star} \in V(\overline{z}_{agt_i})$ and therefore, $\overline{z}_{H_I} \in V(\overline{z}_{agt_I})$.

Furthermore, since the distance between the agent and the same point $\overline{z}_{H_I}$ is decreased, this same point will be chosen by Eq. 28 as the point $\overline{z}_{H_i}$. Note that the index is i, not I in the last sentence. This is because it is possible that the index of the agent that is closest to the set $\tilde{B}_{max}$ may change at the next time step. However, it will change only if another agent moves closer to the set $\tilde{B}_{max}$ than the current agent located at $\overline{z}_{agt_I}$. This does not affect the proof in the sense that the point $\overline{z}_{H_I}$ does not change, simply the index of whichever agent happens to be closest to $\overline{z}_{H_I}$ at the next step. □

Using these results, we can show that the control law with explicit cooperation yields an exhaustive map search

Theorem IV.13. *Under the previously described assumptions and search strategy with explicit cooperation,* $x_w(k, \overline{z}) \to 0 \ \forall \overline{z} \in B$ *(the scores of all cells in the map will approach 0).*

Proof. Theorem IV.10 guarantees that agent I cannot remain in a single cell indefinitely and Theorem IV.11 ensures that it must choose either a cell of non-zero score, $\overline{z}_{h_I}$, or both as the point $\overline{z}_I^\star$ at any given time step. If agent I chooses a cell of non-zero score, Theorem III.1 ensures that the score of that cell is monotonically decreased towards 0.

If the agent does not choose a cell of non-zero score, the only alternative scenario allowed by Theorem IV.11 is that the agent chooses $\overline{z}_{h_I}$ and $x_w(k, \overline{z}_{h_I}) = 0$. By definition of $\overline{z}_{H_I}$, if $x_w(k, \overline{z}_{H_I}) = 0$, then the map has been completely covered since all scores are at most 0. Therefore, assuming that the map has not been entirely searched yet, $x_w(k, \overline{z}_{h_I}) = 0 \Rightarrow \overline{z}_{h_I} \neq \overline{z}_{H_I}$.

Theorem IV.12 ensures that if if $\overline{z}_{h_I} \neq \overline{z}_{H_I}$, choosing $\overline{z}_{h_I}$ as the point $\overline{z}_I^\star$ does not change the value of $\overline{z}_{H_I}$ at the next time step.

At this next time step, the agent once again must choose a cell with non-zero score, $\overline{z}_{h_I}$, or both. If the agent continues to choose $\overline{z}_{h_I}$, eventually $\overline{z}_{H_I} \in \tilde{B}_{R_I}$ and at this point, choosing a cell with non-zero score, $\overline{z}_{h_I}$, or both guarantees that a cell of non-zero score is chosen. Therefore, the score of some cell will be ensured to decrease and given sufficient time, $x_w(k, \overline{z}) \to 0 \ \forall \overline{z} \in B$. □

Remark IV.14. *Theorem IV.13 differs from Theorem III.6 in a few subtle ways. In this situation, the explicit cooperation makes the proof valid only for agent I instead of all agents (Theorem III.6). However, this still guarantees coverage for the team since at any point, there must be an agent in the team which has the properties that allow it to have the index I.*

V. Simulation Results

After analyzing the previously described search algorithm, it can be verified in simulation. A heterogeneous team of agents can be simulated in different environments using the framework described. A heterogeneous team can be modeled by varying parameters in Eq. 10. The parameters used for the different agents using the full algorithm (without explicit cooperation) in are listed in Table 1.

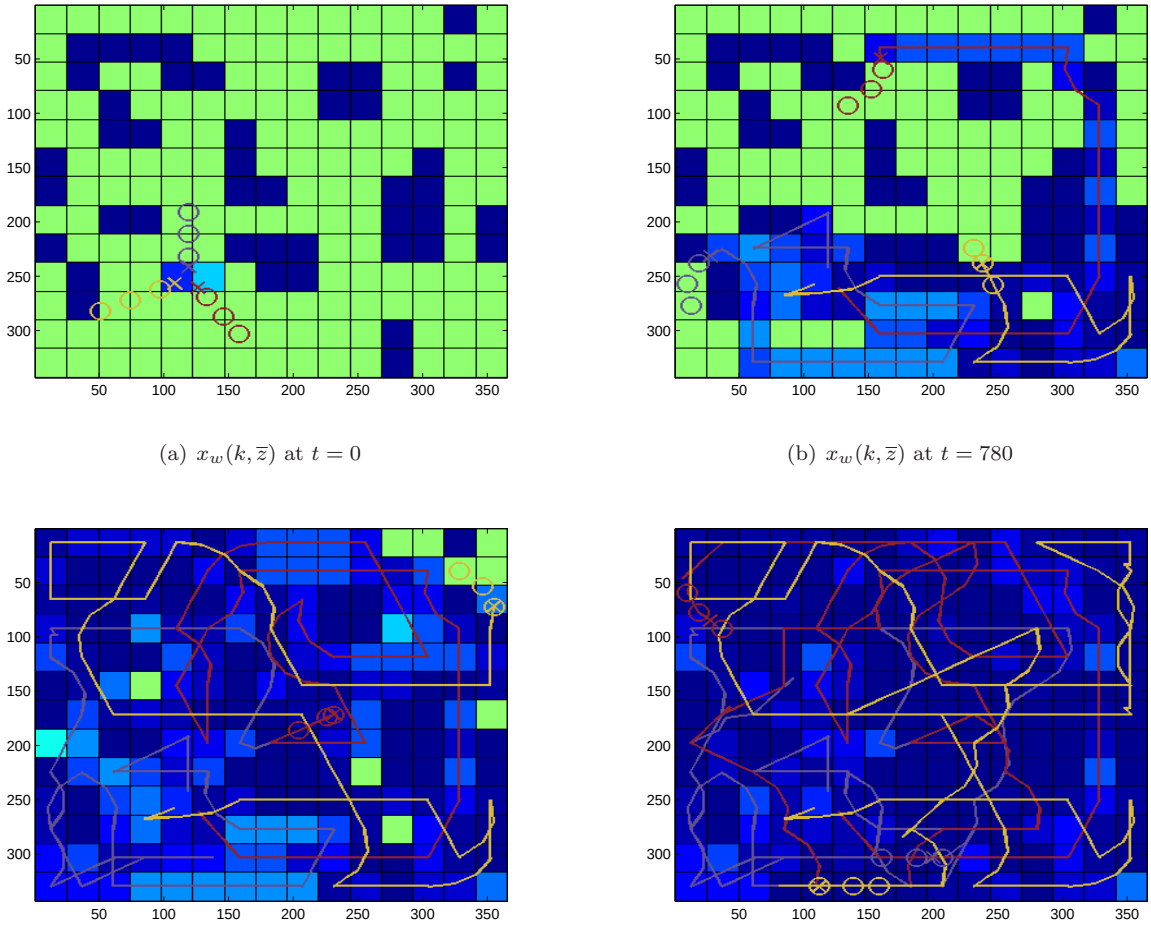
Table 1. Parameters of agents in team during search missions ($d = 3$ for all agents) using full algorithm (no explicit cooperation).

Agent	α	β	γ	δ	h	A.7	$\overline{z}_H$ method
1 (purple)	0.75	0.40	0.95	0.15	0.25	No	Eq. 17
2 (red)	0.90	0.50	0.90	0.05	0.35	No	Eq. 17
3 (gold)	0.10	0.05	0.25	0.35	0.55	Yes	Eq. 17

Simulation results of this test scenario are shown in Figure 9.

The 'x' shows the location of the agent. The 'o' marks illustrate the agent's path, $\overline{w}^\star$, which are the waypoints determined to be the solution to ($\wp_3$). In this situation, the prediction horizon is $d = 3$, so there are 3 'o' marks for each agent. Recall that the only constraint is that the last location of $\overline{w}^\star$ must be equal to $\overline{z}^\star$. Note that only agent 3 (gold) meets the requirement that $\delta > \beta + \gamma$. Having one such agent is sufficient to guarantee complete coverage and eventually the team of agents exhaustively searches the entire map.

In this scenario, the agent's sensors are considerably degraded. Due to their decreased reliability, a single visit to a cell only decreases the score to slightly less than the original score. This can be seen since the cell colors change from green to a light blue instead of a dark blue. Multiple visits to the same cell are required to change the color to dark blue (a score of near 0). Recall the results from Theorem III.6 ensure that the agents visit each cell sufficiently often to drive the scores to zero. Therefore, the algorithm routes the agents to initially search the cells of high probability and then revisits cells as required to drive the scores to zero, guaranteeing an exhaustive search of the map. Additional test cases and analysis[28] have been conducted which support this conclusion.



(a) $x_w(k,\overline{z})$ at $t=0$

(b) $x_w(k,\overline{z})$ at $t=780$

(c) $x_w(k,\overline{z})$ at $t=2080$

(d) $x_w(k,\overline{z})$ at $t=3640$

Figure 9. Full algorithm applied to 3 agents with typical sensors using Eq. 17 to choose $\overline{z}_H$.

A. Scenarios and Performance

To evaluate the performance of the various algorithms, several test scenarios are created. These scenarios are used to emulate situations and environments where a general search algorithm might be used.

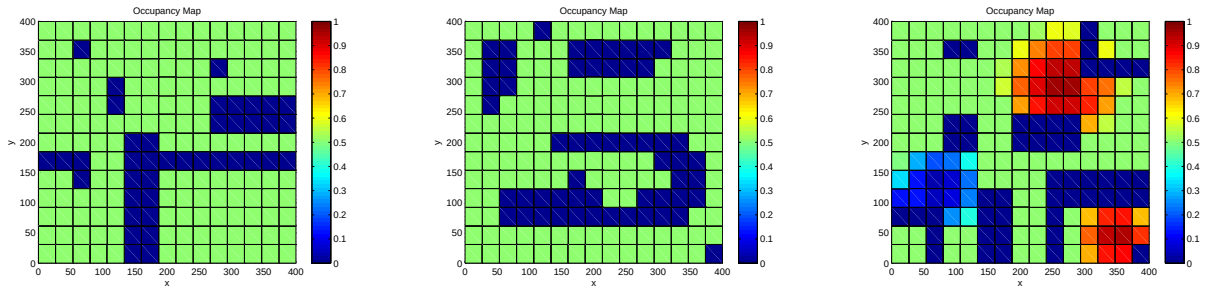
1. *Testing Scenarios*

The algorithm can be tested using several different scenarios. Three representative scenarios are shown in Figure 10. Scenario 1 (S1) is used to represent a possible urban scenario where there are hard obstacles such as buildings to avoid. Scenario 2 (S2) represents a maritime environment where the agents might be searching for a lost ship on the water. Finally, scenario 3 (S3) is another urban environment where there is some a priori knowledge regarding the target's possible location.

2. *Performance Metrics*

In order to gauge performance, several metrics are used. Perhaps the most intuitive is to sum the scores of all the cells. This is directly proportional to the mean of the cell scores. The cumulative map score for a run i at time step k is denoted

$$S(i,k) = \sum_{\overline{z}\in\tilde{B}} x_w(k,\overline{z}) \tag{49}$$



(a) Scenario 1 (S1) (b) Scenario 2 (S2) (c) Scenario 3 (S3)

Figure 10. Several test scenarios used to verify algorithm performance.

Along a similar vein, the variance of the map scores is defined as

$$V(i,k) = \mathrm{Var}(x_w(k,\overline{z})) \text{ for } \overline{z} \in \tilde{B} \tag{50}$$

The average values across n runs are simply

$$S_{ave}(k) = \frac{1}{n}\sum_{i\in I_n} S(i,k) \tag{51}$$

$$V_{ave}(k) = \frac{1}{n}\sum_{i\in I_n} V(i,k) \tag{52}$$

In terms of map coverage, the best and worse case scenarios can be given by

$$S_{max}(k) = \max_{i\in I_n} S(i,k) \tag{53}$$

$$S_{min}(k) = \min_{i\in I_n} S(i,k) \tag{54}$$

In addition to coverage metrics, performance metrics which measure expected time to target detection can be defined. The number of agents which find the target in run i is given by $\tilde{N}(i)$. The average number of agents which find the target in n runs can be defined as

$$\tilde{N}_{ave} = \frac{1}{n}\sum_{i\in I_n} \tilde{N}(i) \tag{55}$$

The time when the m^{th} agent finds the target is given by $T_m(i)$. Note that by definition, $T_1(i) \leq T_2(i) \leq ... \leq T_M(i)$. The average time to target detection for the encounters is slightly more complicated. This is because in some runs, $T_m(i)$ is undefined (in the case that the m^{th} agent does not find the target). Therefore, the values of $T_m(i)$ which contribute to the average are only those when it is defined

$$T_{m,ave} = \frac{1}{\hat{n}(m)}\sum_{i\in I_{\hat{n}(m)}} T_m(i) \tag{56}$$

Where $\hat{n}(m)$ is the number of runs where at least m agents find the target and $I_{\hat{n}(m)}$ corresponds to the indices where the times occur. The ramifications of this definition are explained in the context of the simulation in Section V.C.2.

B. Alternative Search Strategies for Comparison

The searching algorithm can be compared with other common search strategies to evaluate its performance. One of the simplest search strategies is the simple raster scan. This involves the agents moving in a north/south or east/west lines until the boundary of the search is reached. The agent then moves over by one row and then turns around.

Another heuristic search strategy that is related to the raster scan is the "lawn mower" algorithm. It is referred to as a lawn mower algorithm because the agents follow a set of heuristics which might be similar to a person mowing a lawn and involves the agent moving in a straight line until it encounters an obstacle and turning in a direction clear of obstacles.

A third simple search strategy is the gradient climb algorithm. In this case, the algorithm evaluates the scores of the cells surrounding it (to the north, east, south and west) and then chooses the cell which has the highest score. If two or more cells have the same maximum score, the algorithm chooses the cell which requires the smallest course change to visit. This is similar to a steepest ascent algorithm[37,38] where the feasible directions are only the four cardinal directions.

Finally, the Voronoi partitioning method has been used by several groups to generate coverage algorithms which can be applied in this situation[17,16] . In this case, the Voronoi diagram $\overline{V}$ is generated using the agent's positions as generators. Then the point for the agent to search within the next d steps is chosen to be within the agent's Voronoi polygon and its reachable set. This algorithm is outlined in Figure 11.

For purposes of comparison, the full algorithm refers to the algorithm described in Section III (no explicit cooperation between agents) and the full algorithm with Voronoi partitioning refers to the algorithm described in Section IV (algorithm with explicit cooperation between agents).

1 Generate Voronoi diagram of system using agent locations as generators.

2 if $V(\overline{z}_{agt_i}) \bigcap B_{R_i} \neq \emptyset$

3 Choose $\overline{z}_i^\star \in V(\overline{z}_{agt_i}) \bigcap B_{R_i}$

4 else

5 Choose $\overline{z}_i^\star = \overline{z}_{agt_i}$

6 end

7 Generate linear path from $\overline{z}_{agt_i}$ to $\overline{z}_i^\star$

Figure 11. Pseudo code for randomized Voronoi partitioning algorithm.

C. Comparison Results

The various comparison algorithms are used in the three different scenarios. The performance metrics defined previously in Section V.A.2 can then be used to judge each algorithm's efficiency.

1. *Map Coverage*

When evaluating the map coverage by the different strategies, the relevant quantities to analyze are the cumulative map scores and other related metrics. These scenarios involve multiple agents in the team searching the map with no targets.

In order to judge the general behavior of the algorithms, a series of Monte Carlo simulations are used. In these simulations, the performance of each algorithm is gauged over a series of 20 runs and then averaged using Eq. 51. The best and worst case scenarios for the series of runs (in terms of map coverage) are also computed using Eq. 54 and Eq. 53, respectively. The results for scenario 3 are presented in Figure 12 (the traces for the other scenarios display similar trends).

In Figure 12, the solid line represents $S_{ave}(k)$ and the dashed line represents $S_{max}(k)$ for the corresponding search strategy. The performance of the various search strategies using $S_{ave}(k)$ and $S_{max}(k)$ as metrics is summarized in Table 2. In this table, 1 corresponds to the best performance and 6 corresponds to the worst performance.

Looking at average performance measured by $S_{ave}(k)$, it can be seen that in scenarios 1 and 3, the performance of the algorithms from worst to best appears to be, lawn mower, randomized Voronoi, raster scan, gradient climb, full algorithm, then full algorithm with Voronoi partitioning. This is the expected result and shows that the best performance and guarantee of map coverage is achieved with the full algorithm. Furthermore, it shows that the performance is further increased (and the coverage guarantee is preserved) when augmenting the full algorithm with explicit cooperation between agents through the Voronoi partitioning. It should be noted that if the simulation were run for a longer amount of time, it is expected that the raster scan algorithm will eventually outperform the gradient climb when using $S_{ave}(k)$ as the metric for map coverage. Map coverage is guaranteed with the raster scan algorithm, but it is obvious that the performance is suboptimal. Note that in scenario 2, it appears that the gradient climb algorithm performs the best. As mentioned previously, this occurs because the areas to be searched are connected and the

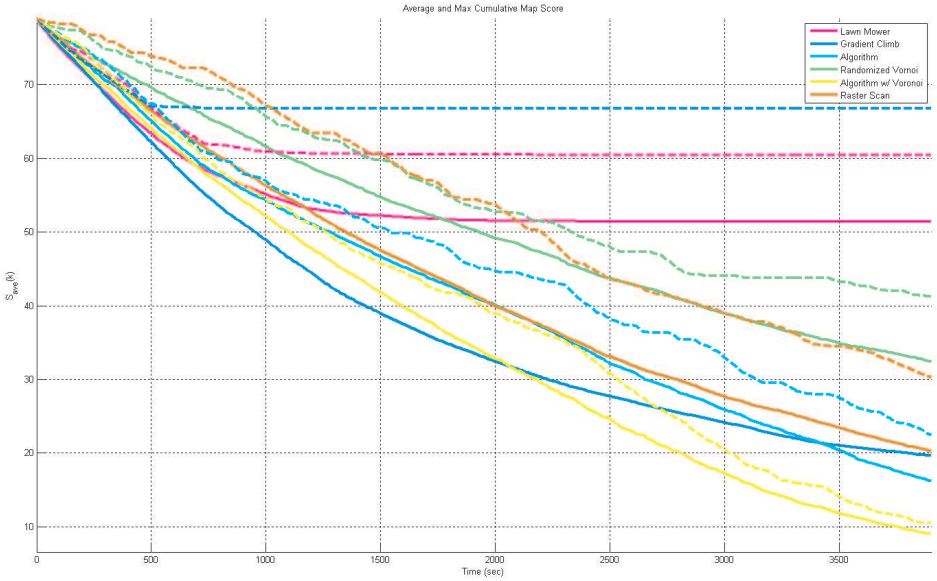


Figure 12. $S_{ave}(k)$ **and** $S_{max}(k)$ **for scenario 3.**

Table 2. Rankings of search strategies using $S_{ave}(k)$ **and** $S_{max}(k)$ **as metrics (1 = best).**

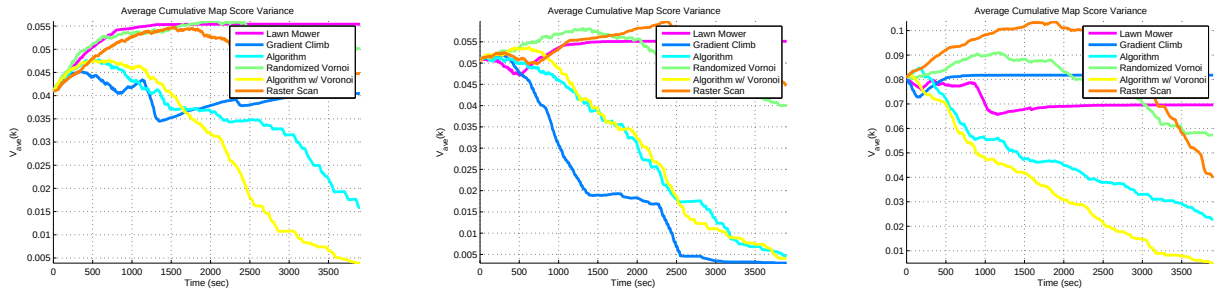
	$S_{ave}(k)$			$S_{max}(k)$		
Strategy	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
Lawn Mower	6	6	6	6	6	5
Randomized Voronoi	5	5	5	4	4	4
Raster Scan	4	4	4	5	5	3
Gradient Climb	3	1	3	3	2	6
Full Algorithm	2	3	2	2	3	2
Full Algorithm Co-op	1	2	1	1	1	1

environment is fairly simple. If the environment was comprised of long, narrow corridors, the gradient climb algorithm would perform poorly due to the fact that it would not cross over areas of low score whereas the full algorithm would.

The guarantees of map coverage are more evident when looking at the worst case scenario for map coverage. Recall that $S_{max}(k)$ is a measure of the worst case scenario possible over all test cases. In this case, it is obvious that the full algorithm with explicit cooperation is the best policy to use. Although the gradient climb strategy may work well for some situations, there are situations where it performs the worst out of all the possible strategies ($S_{max}(k)$ for scenario 3).

The average variance of the runs can be computed using Eq. 52 and the results are shown in Figure 13. These supply further proof that an exhaustive map search is guaranteed only using the raster scan, full algorithm, and full algorithm with explicit cooperation strategies. Of these guaranteed methods, the latter two provide superior performance.

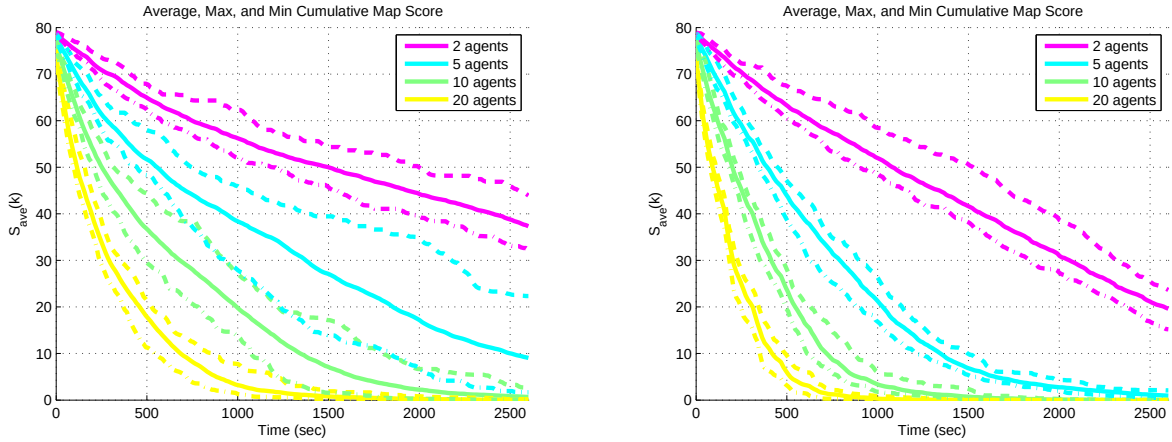
Improving performance in a searching mission typically involves increasing the number of agents involved in the search. The main challenge with current unmanned systems is that raising the number of agents greatly increases operator workload required to manage the team. If the team is comprised of heterogeneous agents with different capabilities, the mission management task becomes even more complicated. The benefits of the



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 13. $V_{ave}(k)$ **for scenarios.**

full algorithm both with and without explicit cooperation within this framework can be seen when looking at the coverage vs. time for varying number of agents. For example, the effects of varying the number of homogeneous agents with the full algorithm and full algorithm with explicit cooperation strategies are shown shown in Figure 14(a) and Figure 14(b), respectively.



(a) Full algorithm (b) Full algorithm with explicit cooperation

Figure 14. Coverage metrics for varying number of agent using scenario 3.

Figure 14 displays several interesting phenomena. First, the guarantee of exhaustive map searching is reinforced. Also, the effect of increasing the number of agents involved in the search is evident since in both cases, the time to drive the map scores to zero decreases as the number of agents increase. The effect of increasing the number of agents in the team can be investigated by calculating the settling time for the coverage metric. In this context, the settling time is defined as the time it takes for $S_{ave}(k)$ to drop below 15% of the initial cumulative map score. The settling time vs. number of agents is shown in Figure 15.

As expected, the settling time decreases as the number of agents increases. Notice that the settling time does not decrease in a linear fashion. Instead, the relative increase in performance decreases with each successive addition of a team member. In other words, adding more agents to the team does not greatly increase performance after a certain point. The trend is roughly exponential and the fitted exponential function is shown with the data as the solid lines. Note that the exponential decay rate for the full algorithm with explicit cooperation is $\lambda = -0.274$ which decays faster than the exponential decay rate for the full algorithm of $\lambda = -0.235$. The modifications to the algorithm to accommodate explicit cooperation benefits the team more as the number of agents in the team increases. This can be attributed to the Voronoi partitioning which helps ensure that agents do not overlap and search the same cells as the mission progresses.

2. Time to Target Detection

There are several parameters that measure the efficiency of the algorithm in terms of target detection time. One metric is the average number of agents that find the target for a given scenario. Related to this metric is the number of scenarios where at least 1, 2, or 3 agents find the target. All runs for the time to target detection Monte Carlo simulation use 200 time steps with 3 agents.

The values of $\tilde{N}_{ave}$, $\hat{n}(1)$, $\hat{n}(2)$, and $\hat{n}(3)$ for 30 runs are summarized in Table 3.

The easiest metric to interpret is the average number of agents that find the target. This is shown in the first three columns of Table 3. As can be seen, the full algorithm performs the best in all scenarios in terms of getting the most agents to find the target. The full algorithm with explicit cooperation does not perform as well for the reasons previously stated. These same reasons explain why the randomized Voronoi search strategy has no runs where more than one agent finds the target.

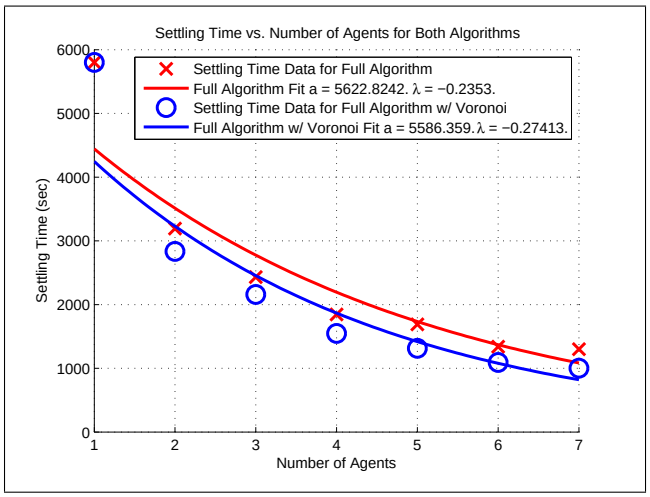


Figure 15. Settling time for various number of agents using full algorithm and full algorithm with explicit cooperation.

It is also worth noting that the raster scan method does not appear to be affected by the different scenarios. This makes sense considering that information about the environment is not taken into account when using this method. It should be noted that in scenario 3, many of the algorithms which use information about the environment (gradient climb, full algorithm, full algorithm with explicit cooperation) show a decrease in performance. This is because the initial world map (shown previously in Figure 10(c)) has several regions of high score. These would correspond to a priori knowledge of possible target locations. Despite this being the initial state of the world, the locations of the targets are placed according to a uniform distribution across the map. In other words, the initial target location is not distributed according to the distribution shown in Figure 10(c). This simulates a situation where the agents are given inaccurate a priori knowledge before starting the mission, thus leading to a decrease in performance. To simulate a situation with accurate information, the initial target location would need to be distributed according to the distribution shown in Figure 10(c).[39]

The columns for $\hat{n}(m)$ show the number of runs where m agents find the target. Once again, it is apparent that methods which use information about the environment perform the best.

Table 3. Average number of agents which find target and number of scenarios where at least 1, 2, or 3 agents finds the target (30 scenarios each).

	Average # Agents Which Find Target $\tilde{N}_{ave}$			# Runs w/ 1 Encounter $\hat{n}(1)$			# Runs w/ 2 Encounters $\hat{n}(2)$			# Runs w/ 3 Encounters $\hat{n}(3)$		
Strategy	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
Lawn Mower	0.267	0.367	0.233	5	8	5	3	2	2	0	1	0
Randomized Voronoi	0.467	0.433	0.633	14	13	19	0	0	0	0	0	0
Raster Scan	1.000	1.033	1.200	20	23	23	9	8	11	1	0	2
Gradient Climb	1.400	2.067	1.500	23	27	19	13	24	16	6	11	10
Full Algorithm	2.567	2.400	1.500	28	24	16	26	24	15	23	24	14
Full Algorithm Co-op	1.833	0.433	0.867	24	29	14	17	18	10	4	8	2

Information regarding the average amount of time required to find the target are displayed in Table 4.

The results in Table 4 are not as good of a representation of performance as those shown in Table 3. The reason can be seen from the definition of $T_{m,ave}$ in Eq. 56. These averages are computed only for the situations where the agents find the target. For example, with the lawn mower strategy with scenario 1, from Table 3, it can be seen only 5, 3, and 0 runs out of 30 show at least 1, 2, or 3 agents finding the target,

Table 4. Average time to target detection by first, second, and third agent (30 scenarios each).

	Average Time for 1st Encounter $T_{1,ave}$			Average Time for 2nd Encounter $T_{2,ave}$			Average Time for 3rd Encounter $T_{3,ave}$		
Strategy	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**	**S1**	**S2**	**S3**
Lawn Mower	143	322	143	500	774	501	N/A	754	N/A
Randomized Voronoi	1114	845	1115	N/A	N/A	N/A	N/A	N/A	N/A
Raster Scan	1045	807	1045	1797	1412	1798	1989	N/A	1989
Gradient Climb	1115	814	1116	1222	1316	1223	1481	1715	1481
Full Algorithm	844	915	844	904	1143	904	1238	1264	1238
Full Algorithm Co-op	1054	1326	979	1526	1470	1791	1703	1370	2015

respectively. Therefore, the averages $T_{1,ave}$, $T_{2,ave}$, and $T_{3,ave}$ are computed using only 5, 3, and 0 samples (explaining why $T_{3,ave}$ = N/A). Using the lawn mower strategy, the agents tend to become stuck in limit cycles and therefore, if the agent is going to find the target, it is will happen very quickly or not at all. This shows why the values of $T_{1,ave}$ and $T_{2,ave}$ are low. This can be compared with the full algorithm where it is virtually guaranteed that the agents will find the target but it will take a longer amount of time to do so.

Furthermore, notice that it is not required that $T_{1,ave} \leq T_{2,ave} \leq ... \leq T_{M,ave}$. For example, for the full algorithm with explicit cooperation using scenario 2, $T_{3,ave} < T_{2,ave}$. Once again, this is because the average is computed over a different number of samples.

VI. Conclusions

Improving performance in a searching mission typically involves increasing the number of agents involved in the search. The main challenge with current manned systems is that raising the number of agents greatly increases operator workload required to manage the team. If the team is comprised of heterogeneous agents with different capabilities, the mission management task becomes even more complicated. This paper presents a modular and scalable searching algorithm that can be used to coordinate a large number of heterogeneous agents involved in a searching mission. The centralized occupancy based map represents the system's belief of the state of the world at a given time. The map can be propagated forward in time to provide an estimate of the future state of the world at step $k+d$. Each agent then decides which coordinate is the most desirable to search in the next d steps. The team can be comprised of different types of agents with different capabilities. The formulation allows each agent to determine what is desirable for its individual capabilities. Each agent then computes control decisions based on the predicted future state of the world. Although these actions may not be optimal in a single step, they will benefit the agent in the future.

Using the full algorithm, there is no explicit cooperation between agents in the team. Instead, the agents are implicitly coupled through the centralized occupancy map. The algorithm remains scalable because each agent does not need to explicitly know about the existence of other agents. Each agent executes the searching algorithm and the resulting emergent behavior is that the team performs a coordinated search. Explicit cooperation between agents is incorporated using the Voronoi partitioning method which simultaneously increases performance and computational costs.

The modularity of the algorithm allows the user to tailor specific parts of the algorithm to address individual agent capabilities. This paper focused primarily on the solution for ($\wp_2$) and showed that under the proposed solution, the agents are shown to perform an exhaustive search of the map regardless of the methods used to provide solutions to ($\wp_1$) and ($\wp_3$).

VII. Further Research

Future work in this area is directed towards increasing searching performance for all agents. Work is directed towards tailoring the reward function and finding additional constraints which may be more relaxed

than assumption A.7 and still guarantees coverage. The issue of system performance is tied tightly to the way that ($\wp_3$) is solved. A closer look at the relationships between ($\wp_2$) and ($\wp_3$) is the focus of current research activity. In addition, the target may be moving or evading the searching agents and therefore, the assumption of a static environment may not be valid. Preliminary investigations of a dynamic environment model have been conducted with promising results[28] but are beyond the scope of this paper. Investigations into incorporating a dynamic world model and changing scores are also being conducted.

VIII. Acknowledgements

This work is sponsored in part by the Washington Technology Center (grants F04-MC2 and F04-MC3), the Osberg Family Trust Fellowship, the Washington NASA Space Grant Consortium, and the Air Force Office of Scientific Research (grant FA-9550-07-1-0528).

References

[1]Monekosso, N. and Remagnino, P., "Robot Exploration Using the Expectation-Maximization Algorithm," *Proceedings of 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Kobe, Japan, 2003.

[2]Dollarhide, R. L., Agah, A., and Minden, G. J., "Evolving Controllers for Autonomous Robot Search Teams," *Artificial Life and Robotics Journal*, Vol. 5, 2002, pp. 178–188.

[3]Kurabayashi, D., Tsuchiya, H., Fujiwara, I., Asama, H., and Kawabata, K., "Motion Algorithm for Autonomous Rescue Agents Based on Information Assistance System," *Proceedings of the 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Kobe, Japan, 2003.

[4]Kantor, G., Singh, S., Peterson, R., Rus, D., Das, A., Kumar, V., Pereiera, G., and Spletzer, J., "Distributed Search and Rescue with Robot Sensor Teams," *Proceedings of the 4th International Conference on Field and Service Robotics*, 2003.

[5]Jennings, J. S., Whelan, G., and Evans, W. F., "Cooperative Search and Rescue with a Team of Mobile Robots," *Proceedings of the International Conference on Advanced Robotics*, 1997.

[6]Benkoski, S. J., Monticino, M. G., and Weisinger, J. R., "A Survey of the Search Theory Literature," *Naval Research Logistics*, Vol. 38, 1991, pp. 469–494.

[7]Bourgault, F., Furukawa, T., and Durrant-Whyte, H., "Coordinated Decentralized Search for a Lost Target in a Bayesian World," *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, Australian Centre for Field Robotics, Las Vegas, NV, October 2003.

[8]Bourgault, F. and Durrant-Whyte, H. F., "Communication in General Decentralized Filters and the Coordinated Search Strategy," *Proceedings of the 7th International Conference on Information Fusion*, Australian Centre for Field Robotics, Stockholm, Sweden, 2004.

[9]Wong, E.-M. and Bourgault, Frederic Furukawa, T., "Multi-vehicle Bayesian Search for Multiple Lost Targets," *Proceedings of the 2005 IEEE Internationalo Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

[10]Polycarpou, M. M., Yang, Y., and Passino, K. M., "A Cooperative Search Framework for Distributed Agents," *Proceedings of the 2001 IEEE International Symposium on Intelligent Control*, Mexico City, Mexico, 2001.

[11]Flint, M., Polycarpou, M., and Fernandez-Gaucherand, E., "Cooperative Control for Multiple Autonomous UAV's Searching for Targets," *Proceedings of the 41st IEEE Conference on Decision and Control*, University of Cincinnati, Las Vegas, NV, 2004.

[12]Jin, Y., Liao, Y., Minai, A. A., and Polycarpou, M. M., "Balancing Search and Target Response in Cooperative Unmanned Aerial Vehicle (UAV) Teams," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 36, 2006, pp. 571–587.

[13]Erignac, C. A., "An Exhaustive Swarming Search Strategy Based on Distributed Pheromone Maps," Tech. rep., Boeing, Seattle, WA, 2004.

[14]Du, Q., Faber, V., and Gunzburger, M., "Centroidal Voronoi Tessellations: Applications and Algorithms," *Society for Industrial and Applied Mathematics Review*, Vol. 41, 1999, pp. 637–676.

[15]Cortes, J., Martinez, S., Karatas, T., and Bullo, F., "Coverage Control for Mobile Sensing Networks," *IEEE Transactions on Robotics and Automation*, Vol. 20, 2004, pp. 243–255.

[16]Laventall, K. and Cortes, J., "Coverage Control By Robotic Networks with Limited-Range Anisotropic Sensory," *Proceedings of the 2008 American Control Conference*, Seattle, Washington, 2008.

[17]Frazzoli, E. and Bullo, F., "Decentralized Algorithms for Vehicle Routing in a Stochastic Time-Varying Environment," *Proceedings of the IEEE Conference on Decision and Control*, December 2004.

[18]Arsie, A. and Frazzoli, E., "Efficient Routing of Multiple Vehicles with No Communication," *Proceedings of the 2007 American Control Conference*, New York City, New York, 2007.

[19]Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulz, D., and Steward, B., "Distributed Multi-Robot Exploration and Mapping," *Proceedings of the 2nd Canadian Conference on Computer and Robot Vision*, 2005.

[20]Baronov, D. and Baillieul, J., "Search Decisions for Teams of Automata," *Proceedings of the 47th Conference on Decision and Control*, Cancun, Mexico, 2008.

[21]Hoffman, G. M., Waslander, S. L., and Tomlin, C. J., "Distributed Cooperative Search Using Information-Theoretic Costs for Particle Filters, with Quadrotor Applications," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Keystone, CO, 2006.

[22]Rubio, J. C., Vagners, J., and Rysdyk, R. T., "Adaptive Path Planning for Autonomous UAV Oceanic Search Missions," *Proceedings of the 1st AIAA Intelligent Systems Technical Conference*, 2004.

[23]Lum, C. W., Rysdyk, R. T., and Pongpunwattana, A., "Occupancy Based Map Searching Using Heterogeneous Teams of Autonomous Vehicles," *Proceedings of the 2006 Guidance, Navigation, and Control Conference*, Keystone, CO, August 2006.

[24]Lum, C. W. and Rysdyk, R. T., "Time Constrained Randomized Path Planning Using Spatial Networks," *Proceedings of the 2008 American Control Conference*, Seattle, WA, June 2008.

[25]Elfes, A., "Using Occupancy Grids for Mobile Robot Perception and Navigation," *IEEE Computer*, 1989, pp. 46–57.

[26]Elfes, A., *Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, May 1989.

[27]Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

[28]Lum, C. W., *Coordinated Searching and Target Identification Using Teams of Autonomous Agents*, Ph.D. thesis, University of Washington, Seattle, WA, March 2009.

[29]Lum, C. W. and Vagners, J., "A Modular Algorithm for Exhaustive Map Searching Using Occupancy Based Maps," *To appear in Proceedings of the 2009 Infotech@Aerospace Conference*, Seattle, WA, April 2009.

[30]Lum, C. W. and Rysdyk, R. T., "Feature Extraction of Low Dimensional Sensor Returns for Autonomous Target Identification," *Proceedings of the 2008 Guidance, Navigation, and Control Conference*, Honolulu, HI, August 2008.

[31]Carson, J. M. I. and Ackmese, B., "A Model Predictive Control Technique with Guaranteed Resolvability and Required Thruster Silent Times for Small-Body Proximity Operations," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Keystone, CO, 2006.

[32]Pongpunwattana, A., *Real-Time Planning for Teams of Autonomous Vehicles in Dynamic Uncertain Environments*, Ph.D. thesis, University of Washington, Seattle, WA, June 2004.

[33]Klein, D. J., *Coordinated Control and Estimation for Multi-agent Systems: Theory and Practice*, Ph.D. thesis, University of Washington, Seattle, WA, September 2008.

[34]Lalish, E., Morgansen, K. A., and Tsukamaki, T., "Decentralized Reactive Collision Avoidance for Multiple Unicycle-Type Vehicles," *Proceedings of the 2008 American Control Conference*, Seattle, WA, 2008.

[35]Nair, S. and Marsden, J., "Collision Avoidance and Surveillance Measures for Multivehicle Systems," Tech. rep., California Institute of Technology, Pasadena, CA, 2008.

[36]Okabe, A., Boots, B., and Sugihara, K., *Spatial Tessellations Concepts and Applications of Voronoi Diagrams*, John Wiley and Sons, 1996.

[37]Boyd, S. and Vandenberghe, L., *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[38]Rockafellar, R., "Fundamentals of Optimization," Tech. rep., University of Washington, Seattle, WA, 2006.

[39]Thrun, S., Burgard, W., and Fox, D., *Probabilistic Robotics*, MIT Press, 2005.